HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

foryouandyourcustomers

Modellierung und Visualisierung von komplexen Produktstrukturen

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2012

Autoren:	Christoph Rosenberger
	Michael Steiner
Betreuer:	Dr. Daniel Keller
Projektpartner:	foryouandyourcustomers, Pfäffikon ZH
Experte:	Dr. Rudolf Mattmann
Gegenleser:	Prof. Hans Rudin

Abstract

PIM-Software wird von verschiedenen Anbietern entwickelt. Trotzdem gibt es noch kein Tool, welches bei der Modellierung von Produktstrukturen Unterstützung bietet. In dieser Bachelorarbeit haben wir einen Prototyp erstellt, welcher helfen soll, diese Lücke zu schliessen.

Wir haben ein sinnvolles Set an Daten identifiziert, welche in der ersten Phase einer Produktstrukturierung erfasst werden müssen:

- Attribute welche die Produkte charakterisieren
- Klassen um Produkte zu abstrahieren und mit Attributen zu beschreiben
- Vererbungsbeziehungen zwischen den Klassen um die Produkte zu strukturieren
- Metaattribute zu Attributen und Klassen, diese müssen für die verschiedenen Produktstrukturierungsprojekte angepasst werden
- Enumeration mit ihren Werten (z.B. ein Farbraum mit allen zugehörigen Farbedefinitionen)

Um die Komplexität und den Umfang einer Produktstruktur zu bewältigen, bieten wir verschiedene Visualisierungen an. Die wichtigsten sind:

- Eine einfache Auflistung aller Attribute und Klassen inklusive der zugehörigen Metaattributen.
- Die komplette Vererbungshierarchie wird in einer übersichtlichen Baumansicht einfach visualisiert.
- Alternativ lässt sich die Vererbungshierarchie als UML darstellen.
- Für die einzelnen Produkte lassen sich Eingabemasken erzeugen, welche einen Eindruck über den Umfang der zu erfassenden Daten und den damit einhergehenden Aufwand bieten.

Ausserdem lässt sich durch Setzen von Filtern und Markierungen die Produktstruktur explorativ begreifen.

foryouandyourcustomers [1]

Über foryouandyourcustomers

foryouandyourcustomers ist ein junges Unternehmen, welches sich im Bereich Multichannel stark gemacht hat. Sie unterscheiden sich enorm von Firmen, welche einfache E-Commerce Lösungen anbieten. Zur Zeit zählt die Firma 13 Mitarbeiter und befindet sich in einer Hyperwachstumsphase. Sie hebt sich von anderen Firmen ab, weil sie die “Werbesicht” und das technische Know-How liefern. Bei foryouandyourcustomers ist Business und Technologie unter einem Dach.

Ziele

Das Ziel ist es, den Kunden sowie den Endkunden (Konsumenten) zu unterstützen.
Gemeinsam mit dem Kunden untersucht man was für eine Lösung er benötigt, wie umfassend sein PIM sein soll, welche Produkte und welche Attribute darin verwaltet werden sollen, welche Kanäle sinnvoll sind und welche Auswirkungen dies auf die Datenhaltung hat. Des Weiteren wird analysiert, was schon vorhanden ist und wie man diese Daten weiterhin nutzen kann.
Den Endkunden möchte man beim Finden oder dem Vergleich von Produkten unterstützen. Durch Schaffung von Klarheit und Einheitlichkeit auf allen Kanälen kann dies gewährleistet werden. Eine reine Informatik-Firma würde eine gute technische Lösung liefern und eine Werbefirma ein schönes und ansprechendes Design. foryouandyourcustomers bietet dem Kunden beides. Somit ist gewährleistet, dass sich zum Beispiel das Design einer Website technisch auch sinnvoll umsetzen lässt.

Ist - Zustand

Ein Product Information Management System ist sehr komplex und umfangreich, deshalb gehen Details schnell vergessen. Zur Zeit wird dem durch ein möglichst kleines Team entgegengewirkt. Dies führt aber dazu, dass ein Mitarbeiter nicht einfach ausgewechselt werden kann. Wenn ein Mitarbeiter ausgewechselt würde, müsste man wieder von vorne beginnen. Auch ist es schwierig jemand neues einzuweisen, denn das komplette Wissen ist nur in den Köpfen der Mitarbeiter. Eine spätere Weiterentwicklung des Projektes ist somit auch nicht trivial. Das Datenmodell wird direkt im PCM designed.

[1] Übernommen aus unsrer Studienarbeit

Produktstrukturierung

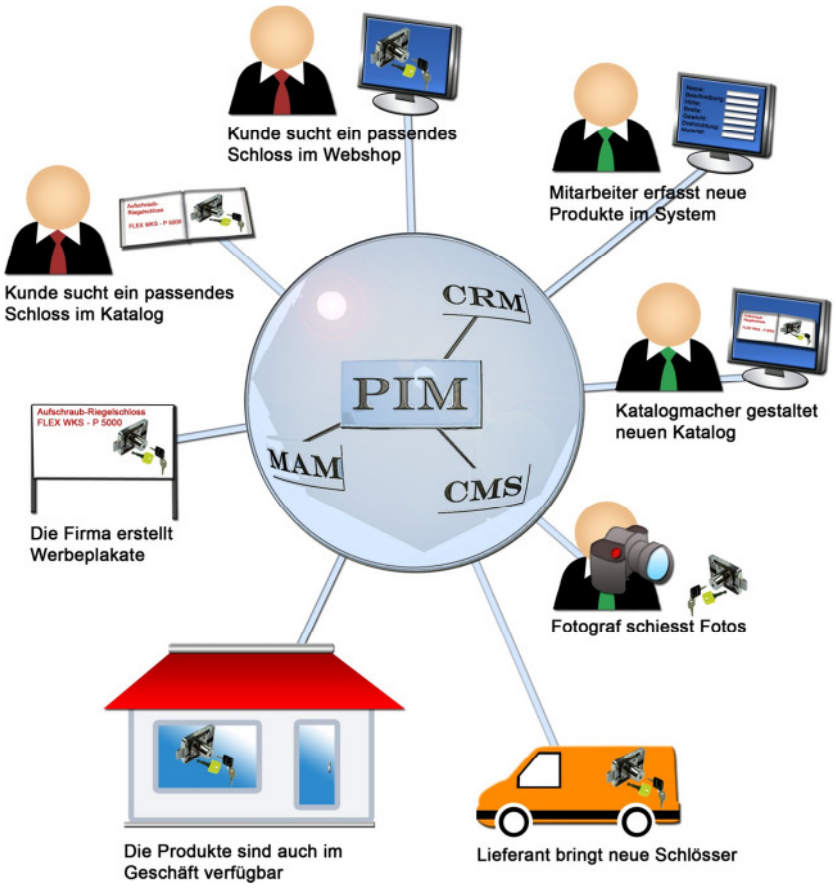


Abbildung 1: Orientierung im Universum

Firmen mit umfangreichem Sortiment stehen vor der Herausforderung, all ihre Produkte ihren Kunden auf verschiedenen Kanälen – Webshop, Katalog, Werbung, etc. – einheitlich zu präsentieren. Um dies zu erreichen, braucht es eine zentrale Verwaltung der Produktdaten: Das Product Information Mangement system PIM. Natürlich muss dies möglichst günstig sein. Um ein System zu installieren, welches alle diese Anforderungen erfüllt, müssen folgende Fragen geklärt werden:

- Welche Produkte gibt es
- Was sind wichtige Informationen die zu jedem Produkt gespeichert werden müssen
- Welche Produkte haben die gleichen Informationen
- Wie hängen die Produkte zusammen (Zubehör, Variation, Upgrade,…)
- Wer ist das Zielpublikum, welche Bedürfnisse hat der Endkunde
- Welches sind die Touchpoints, welches die Customer Journeys
- Wie navigiert der Kunde durch die Produkte
- Nach welchen Informationen muss der Endkunde suchen können
- Und vieles mehr

In einer Produktstrukturierung geht es darum, genau diese Fragen zu beantworten. So kann dann eine Datenstruktur, welche es erlaubt die Vielfalt an Produkten zu speichern und ein System, welches die Bedürfnisse des Anbieters sowie des Kunden abdecken, erstellt werden.

Christoph Rosenberger
Michael Steiner

Inhalt

A. Kurzfassung

A.1 Management Summary

Ausgangslage

Firmen mit einem umfangreichen Sortiment stehen vor der Herausforderung, ihre tausenden von Produkten und sie charakterisierenden Attribute in einer Produktstruktur zu beschreiben. Diese Struktur bildet die Grundlage für die digitale Verwaltung der Produkte und die Erbringung von inhaltlich sauber aufeinander abgestimmten Dienstleistungen wie Webshop, Katalog und Werbung. Für diese komplexen Modellierungen gibt es noch keine strukturierte Vorgehensweise. Weder sind die relevanten Daten identifiziert, noch gibt es ein einheitliches Format um die Resultate zu dokumentieren; anstatt dessen werden Attributlisten und Vererbungsbäume in Excel oder anderen Adhoc-Formaten festgehalten.

Vorgehen

Zu Beginn der Arbeit entwickelten wir Soll-Scenarios und Wireframes. Anschliessend setzten wir unsere Ideen in einem «fertigen» Programm mit kompletter Navigationsstruktur, ersten implementierten Features und integrierten Wireframes um. Im Verlauf des Projekts konnten wir iterativ die bestehenden Ergebnisse diskutieren, den konkreter formulierten Bedürfnissen des Kunden anpassen und Schritt für Schritt weitere nützliche Features implementieren.

Ergebnis

Wir haben ein sinnvolles Set an Daten identifiziert, welche in der ersten Phase einer Produktstrukturierung erfasst werden müssen:

- Attribute welche die Produkte charakterisieren
- Klassen um Produkte zu abstrahieren und mit Attributen zu beschreiben
- Vererbungsbeziehungen zwischen den Klassen um die Produkte zu strukturieren
- Metaattribute zu Attributen und Klassen, diese müssen für die verschiedenen Produktstrukturierungsprojekte angepasst werden
- Enumeration mit ihren Werten (z.B. ein Farbraum mit allen zugehörigen Farbdefinitionen)

Um die Komplexität und den Umfang einer Produktstruktur zu bewältigen, bieten wir verschiedene Visualisierungen an. Die wichtigsten sind:

- Eine einfache Auflistung aller Attribute und Klassen inklusive der zugehörigen Metaattributen.
- Die komplette Vererbungshierarchie wird in einer übersichtlichen Baumansicht einfach visualisiert.
- Alternativ lässt sich die Vererbungshierarchie als UML Klassendiagramm darstellen.
- Für die einzelnen Produkte lassen sich Eingabemasken erzeugen, welche einen Eindruck über den Umfang der zu erfassenden Daten und den damit einhergehenden Aufwand bieten.

Ausserdem lässt sich durch Setzen von Filtern und Markierungen die Produktstruktur explorativ begreifen.

A.2 PSA

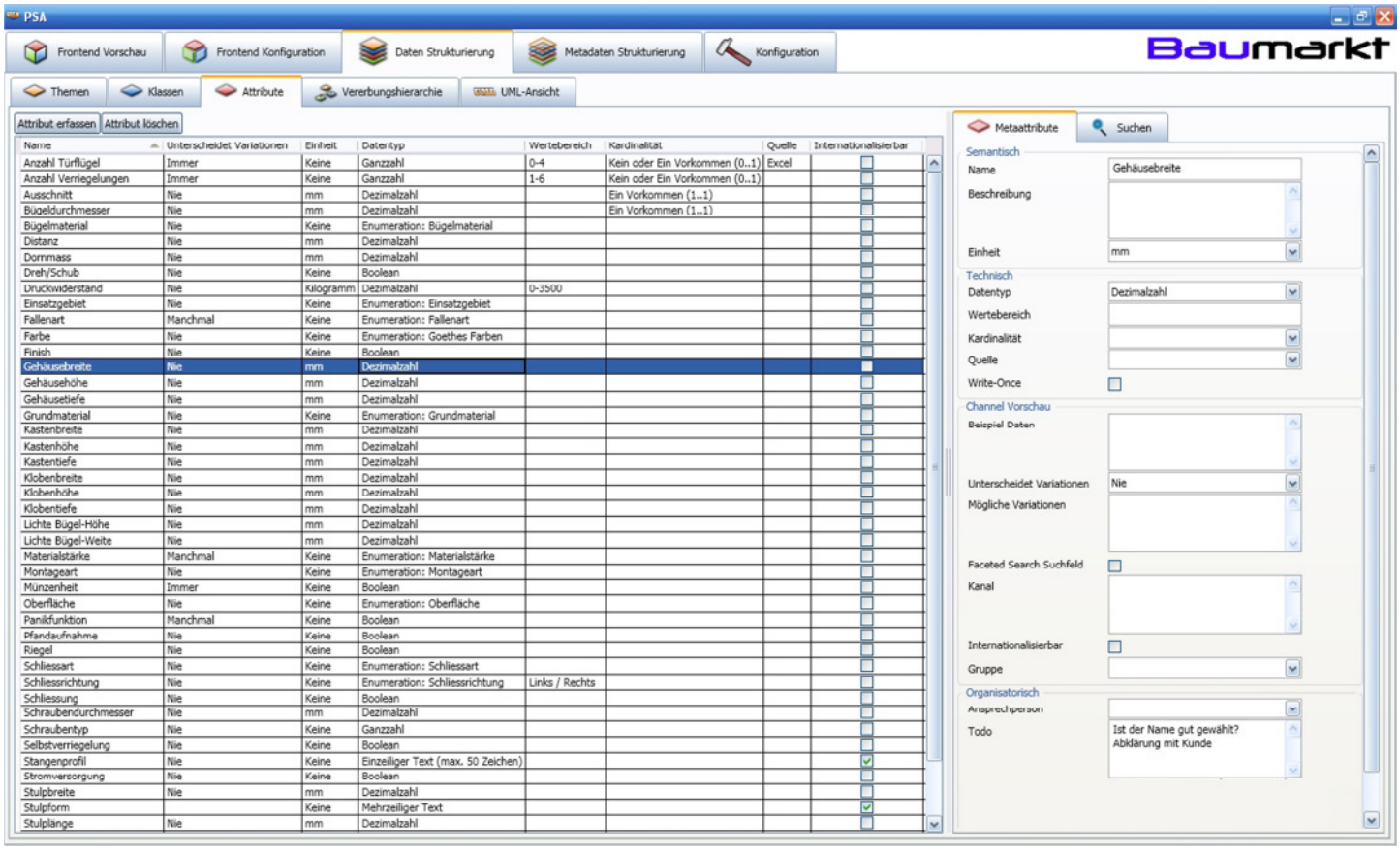


Abbildung 2: Screenshot Attribut Liste

Attribute und Metaattribute

In dieser Ansicht können bequem Attribute gesammelt und verwaltet werden. Durch die zuvor angelegten Metaattribute, kann jedes Attribut genauer beschrieben werden. Dies ist ein enormer Mehrwert: So ein System hat sehr viele Attribute, da kann es schnell einmal zu Namenskonflikten kommen. Durch die Metaattributierung kann sichergestellt werden, dass alle Beteiligten unter einem Attribut dieselbe Bedeutung verstehen.

Auch kann man für jede Information speichern, ob die Daten schon in einem älteren System vorhanden sind oder nicht und ob die Informationen zum Beispiel in eine andere Sprache übersetzt werden müssen. Auch technische Aspekte können hier einfach hinterlegt werden.

Unsere Software bietet ein Standardset an Metaattributen, welche für die meisten Anwendungsfälle eine gute Basis bietet. Natürlich kann der Anwender die Metaattribute auch beliebig erweitern oder verändern, so dass er die für ihn wichtigen Informationen einheitlich und zentral ablegen kann.

Suchen und Filtern

Dank der Such- und Filtermöglichkeiten kann sich der Benutzer schnell einen Überblick verschaffen. So findet er sich auch in umfangreichen Projekten gut zurecht.

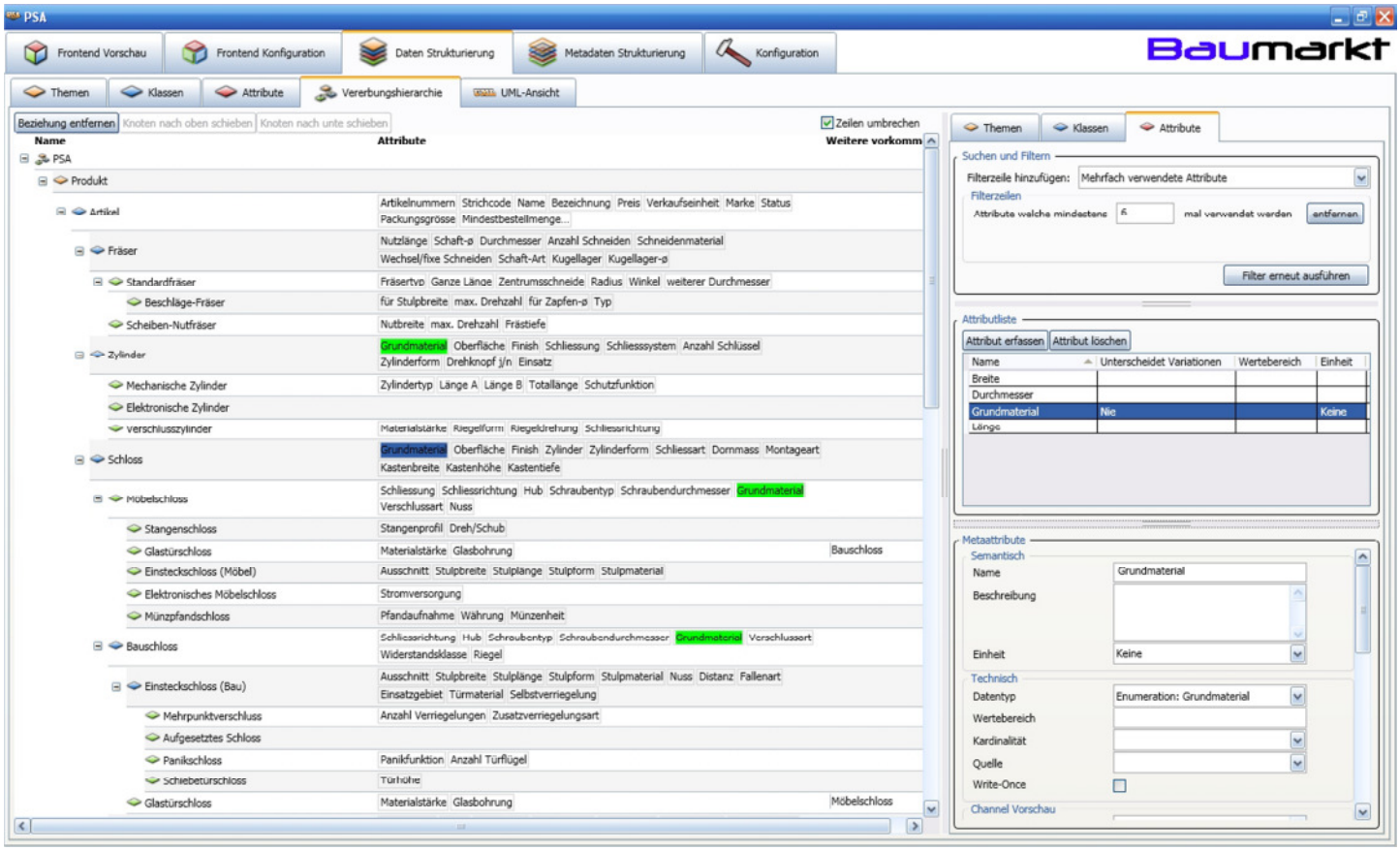


Abbildung 3: Screenshot Vererbungshierarchie

Vererbungs-hierarchie

Das Herzstück von PSA ist die Vererbungshierarchie. Hier kann der Anwender all seine Produkttypen erfassen und ihnen die entsprechenden Attribute bequem per Drag und Drop zuweisen. Auch hier ist es entscheidend, dass alle wichtigen Informationen auf einen Blick sichtbar sind. Deshalb werden die Metaattribute gleich mit angezeigt.

Suchen und Filtern

Dank der Thematisierung, diversen Filtern und guten Suchfunktionen kann auch in einer sehr grossen Modellierung Ordnung und Übersichtlichkeit geschaffen werden. Mit wenigen Klicks werden alle Attribute welche mehrfach verwendet werden markiert. Nicht verwendete Klassen werden schnell sichtbar gemacht und Attribute, welche noch keiner Klasse oder Produkttypdefinition zugewiesen wurden, können angezeigt werden. Natürlich ist die Kombination von mehreren Filtern möglich.

Markieren

Mit dem Filter „Mehrfach verwendete Attribute“ kann man Schwächen in der Modellierung erkennen. Wie im Screenshot ersichtlich, wird jedes Vorkommen des selektierten Attributs im Baum markiert. So kann entschieden werden, ob das Attribut besser einer gemeinsamen Elternklasse zugewiesen würde. Somit enthalten die einzelnen Produttypdefinitionen nur noch jene Attribute, welche sie auch explizit voneinander unterscheiden.

Übersichtlich

Für Diskussionen im Team kann PSA mit dem Beamer projiziert werden. Die Baumstruktur ist sehr schlank und übersichtlich. Sie erlaubt grosse Teile einer Modellierung gleichzeitig darzustellen.

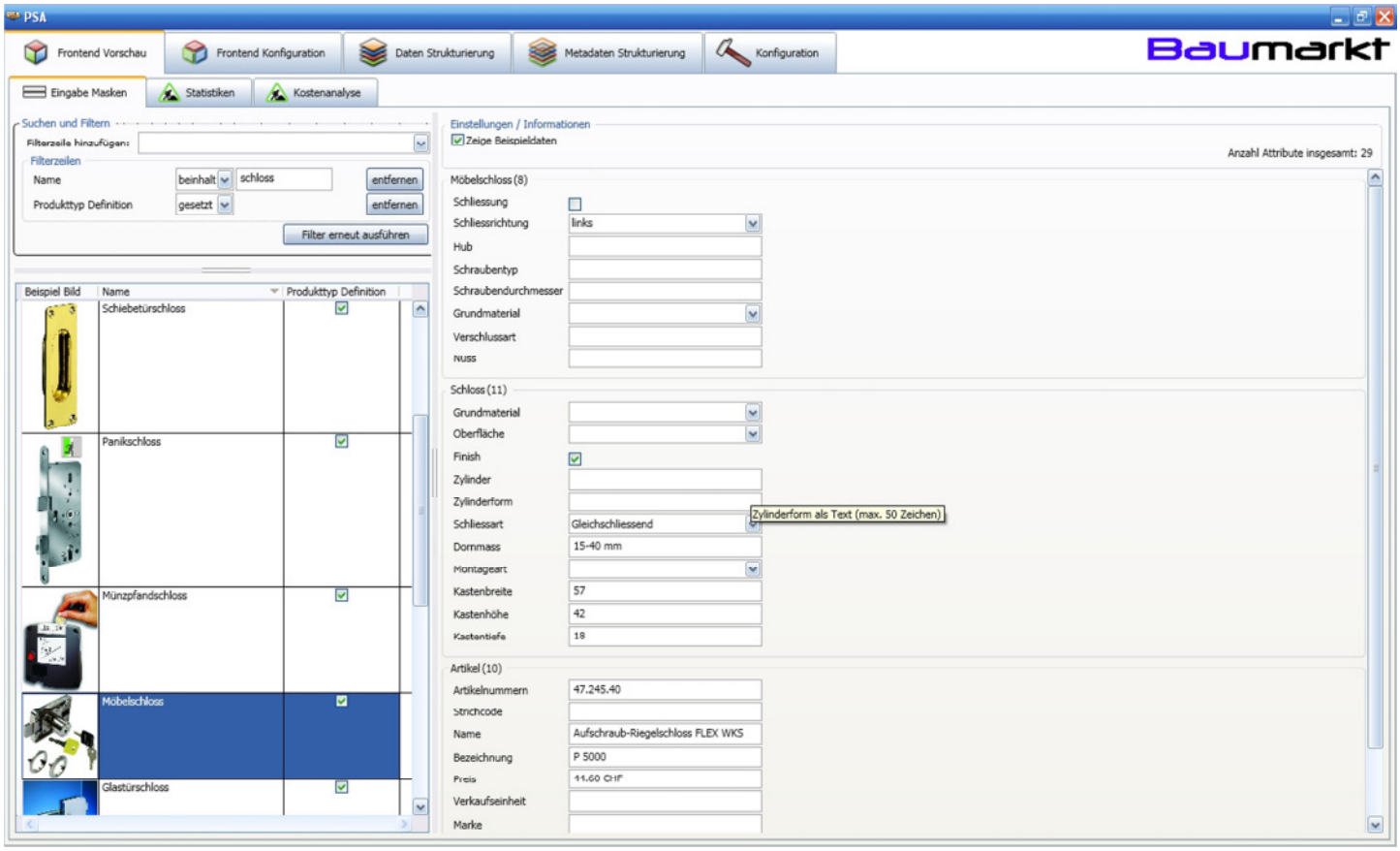


Abbildung 4: Screenshot Eingabe Masken

Produktivsystem Vorschau

Diese Ansicht zeigt die komplette Modellierung einer Klasse oder eines Produkttyps konkret anhand einer Eingabemaske, welche später in einem Produktivsystem sehr ähnlich aussehen wird. Diese Darstellung wird auch sofort von Mitarbeitern verstanden, welche keine Erfahrung mit abstrakten Modellierungen haben. Sie ist sehr konkret und bietet ein vertrautes Bild.

Realitätsnah

Damit die Darstellung möglichst realistisch ist, werden verschiedene Datentypen verschieden abgebildet. So kann man unterscheiden ob man später im Produktivsystem einen Text eingeben oder nur eine Auswahl aus einer Enumeration treffen muss.

Auf Wunsch können Beispieldaten angezeigt werden. Ausserdem hat jedes Attribut einen Tooltipp.

Kostenschätzung

Es ist auf den ersten Blick ersichtlich, welche und vor allem wie viele Daten in dem zukünftigen System eingegeben werden müssen. Dies ermöglicht eine Aufwandschätzung und so kann gut darüber diskutiert werden, ob das Kosten-Nutzen Verhältnis stimmt.

B. Projektbericht

B.1 Ausgangslage

B.1.1 Problemstellung

PCM

Die Technologien welche ein Multichannel Business erst ermöglichen, haben sich in den letzten Jahren rasant entwickelt. Heutige PCM bieten viele Möglichkeiten um die zentral verwalteten Produktdaten auf verschiedenen Kanälen kundengerecht anzubieten. Die Konfiguration dieser Systeme mit XML Dateien ist allerdings noch sehr umständlich. So ist es auch schwierig eine Übersicht über die gemachte Modellierung zu behalten.

Excel

Mangels Alternativen wird heute mit Excel gearbeitet. Einfache Auflistungen von Attributen zu erfassen ist kein Problem. Auch Enumerationen für Metaattribute könnten hinterlegt werden. Die Zuweisung von Attributen zu Klassen oder gar die Erstellung von Vererbungshierarchien bedürfen schon einiger Disziplin um ein einheitliches Dokument zu erstellen; eine Überarbeitung wird dann noch schwieriger. Und spätestens wenn die Daten über das ganze Projekt hinweg synchronisiert sein sollen, hat man mit Excel ein Problem.

Kundenwunsch

Mit der heutigen Situation ist unser Auftraggeber nicht zufrieden. Wie genau eine Lösung aussehen sollte, wusste er allerdings auch nicht. Er wünschte sich eine Software mit welcher man komplexe Produktstrukturen modellieren und visualisieren kann.

Seine Erwartung an diese Bachelorarbeit ist ein Prototyp in welchem Modellierungs- und Visualisierungsmöglichkeiten aufgezeigt und ausprobiert werden können. Dieser Prototyp sollte auch als Diskussionsgrundlage dienen, wie eine „optimale“ Lösung seines Problems aussehen sollte.

B.1.2 Ausgangslage für das Projekt

Studienarbeit

Im letzten Semester haben wir unsere Studienarbeit unter dem Titel „Visualisierung von komplexen Produktstrukturen“ geschrieben. Es war eine eher konzeptionelle Arbeit. Wir untersuchten ein breites Spektrum des Multichannel Universums von den Datenströmen zwischen den Umsystemen und dem PIM, über die Channels bis hin zu den Touchpoints. Wir versuchten komplexe Zusammenhänge einfacher darzustellen durch Nutzung von 3D Darstellungen. Ausserdem zeigten wir erste Verwendungsmöglichkeiten von Metadaten auf.

User Interface 2 Miniprojekt

Zu Beginn dieses Semesters haben für das Modul User Interface 2 ein Miniprojekt durchgeführt. Mit Soll-Scenarios und Wireframes haben wir eine erste Anforderungsanalyse durchgeführt und einen ersten, einfachen Papierprototypen von PSA erstellt.

Die komplette Dokumentation des Miniprojekts ist im Anhang zu finden.

Invisible moving Target

Da wir mit unserer Arbeit Neuland beschreiten, konnte man uns nicht sagen, wie unsere Software ausschauen könnte oder was sie leisten müsste. Ausserdem hatte unsere einzige Ansprechperson bis anhin noch nie in einem Produktstrukturierungsprojekt mitgearbeitet; Herr Keller hat während dem Verlauf unserer Bachelorarbeit erstmals an Sitzungen eines entsprechenden Projekts teilgenommen. Somit blieben viele unserer Fragen unbeantwortet.

Aufgrund der heutigen Situation gibt es auch keine sauber dokumentierten Projekte in diesem Gebiet. Ausserdem ist unser Projektpartner foryouandyourcustomers eine sehr junge Firma: Sie hat bisher noch keine Produktstrukturierung durchgeführt. Da es die rechtliche Situation nicht erlaubt, konnten uns die Mitarbeiter von foryouandyourcustomers auch keine Unterlagen aus Projekten liefern, bei welchen sie für frühere Arbeitgeber mitgewirkt hatten.

Entsprechend der Ausgangslage änderten sich auch immer wieder die Wünsche und Prioritäten während des Projekts. Einige Punkte in denen die Anforderungen änderten seien hier aufgelistet:

- Strukturierung des Prozesses vs. dem User sämtliche Freiheiten bei der Modellierung lassen.
- Proof of Concept um Ideen zu entwickeln und zeigen vs. Einsatzfähiger Prototyp
- Den Metaattributen wurde zu Beginn der Arbeit grosser Stellenwert eingeräumt. Nach einer ersten einfachen – zugegebener massen aber schon sehr flexiblen – Implementierung, waren die Metaattribute kaum mehr ein Thema.

B.1.3 Anforderungsanalyse

B.1.3.1 Vorgehen

Initiale Anforderungen

Unser Auftraggeber hatte nur eine vage Vorstellung von seinen Bedürfnissen oder davon, welche Features PSA bieten könnte. Somit war eine vertiefte Anforderungsanalyse in den ersten Iterationen kaum möglich. Wir mussten während des ganzen Projekts auf wechselnde Anforderungen reagieren.

Fortwährende Anforderungs-analyse

Um trotzdem einen möglichst hohen Kundennutzen zu erzielen, haben wir versucht mit möglichst geringem Aufwand Feedback vom Auftraggeber zu provozieren. Wie im folgenden Abschnitt „User Interface Miniprojekt“ beschrieben, haben wir mit einem reinen Papierprototypen begonnen. Im weiteren Verlauf des Projekts haben wir ein laufendes Programm mit vollständiger Navigationsstruktur erstellt, bei welchem erste Features implementiert waren; den Rest haben wir durch eingebaute Wireframes simuliert. So konnten wir Schritt für Schritt dem Auftraggeber helfen, seine Bedürfnisse zu konkretisieren.

User Interface Miniprojekt

Im Miniprojekt des Moduls User Interfaces 2 haben wir die ersten Anforderungen an PSA analysiert. Mit einer Beschreibung des Ist-Zustandes und mit Soll-Szenarien haben wir erste Erkenntnisse gesammelt. Diese haben wir dann in einem Wireframe Prototypen umgesetzt. Mit Herrn Keller als Testperson haben wir diese erste Version von PSA in einem ca. halbstündigen Papierprototyptest validiert. Durch dieses Vorgehen hatten wir eine solide Basis um mit der Implementierung von PSA zu beginnen.

Die komplette Dokumentation des Miniprojekts ist im Anhang zu finden.

B.1.3.2 Nicht funktionale Anforderungen

Prototyp

Der Auftraggeber erwartet einen funktionierenden Prototyp. Die Anforderungen an Benutzbarkeit / Komfort, Zuverlässigkeit, Effizienz stehen nicht im Zentrum.

Single Desktop Single User

PSA ist eine Single Desktop Single User Applikation. Weder Login noch Netzwerkfunktionalität ist gefordert. Es sind keine grossen Mengen an Daten zu erwarten (ohne Bilder ein paar Megabyte). Die erforderliche Infrastruktur sollte entsprechend einfach sein.

Änderbarkeit

Über die funktionalen Anforderungen war zu Beginn des Projekts nur sehr wenig bekannt. Es war nicht vorhersehbar welche Views und Features zu implementieren waren, noch wie diese zusammenspielen sollten. Somit war die Anforderung an die Änderbarkeit sehr gross.

Technologie

Der Auftraggeber wünschte als Plattform Java oder C# auf .Net. Eine dynamisch typisierte Sprache schloss er aus.

Bei dieser Auswahl sahen wir keine technisch entscheidenden Argumente. Die Wahl für C# ist somit eher durch persönliche Präferenzen entstanden:

- WPF und MVVM schienen uns moderne Ansätze. Bei Java haben wir erst mit Swing erste Erfahrungen gesammelt.

- LINQ ermöglicht auf einfache Weise ausdrucksstarke kurze Programmabschnitte zu schreiben.

B.1.4 Technologien

PSA ist mit C# implementiert.

Das GUI haben wir mit WPF realisiert.

Für den GUI Layer haben wir das MVVM Light Toolkit von Galasoft eingesetzt.

http://www.galasoft.ch/mvvm/

Als OR-Mapper kommt Entity Framework zum Einsatz.

Die Daten werden mit der In-File Datenbank SQL Server Compact persistiert.

Log4Net ist die eingesetzte Logging Library.

Unsere IDE war Visual Studio 2010 Ultimate.

Für die Modellierung von Klassendiagrammen haben wir UMLet eingesetzt.

Die Wireframes haben wir mit balsamiq erstellt.

B.2 Modellieren und Visualisieren mit PSA

B.2.1 Vererbungshierarchie

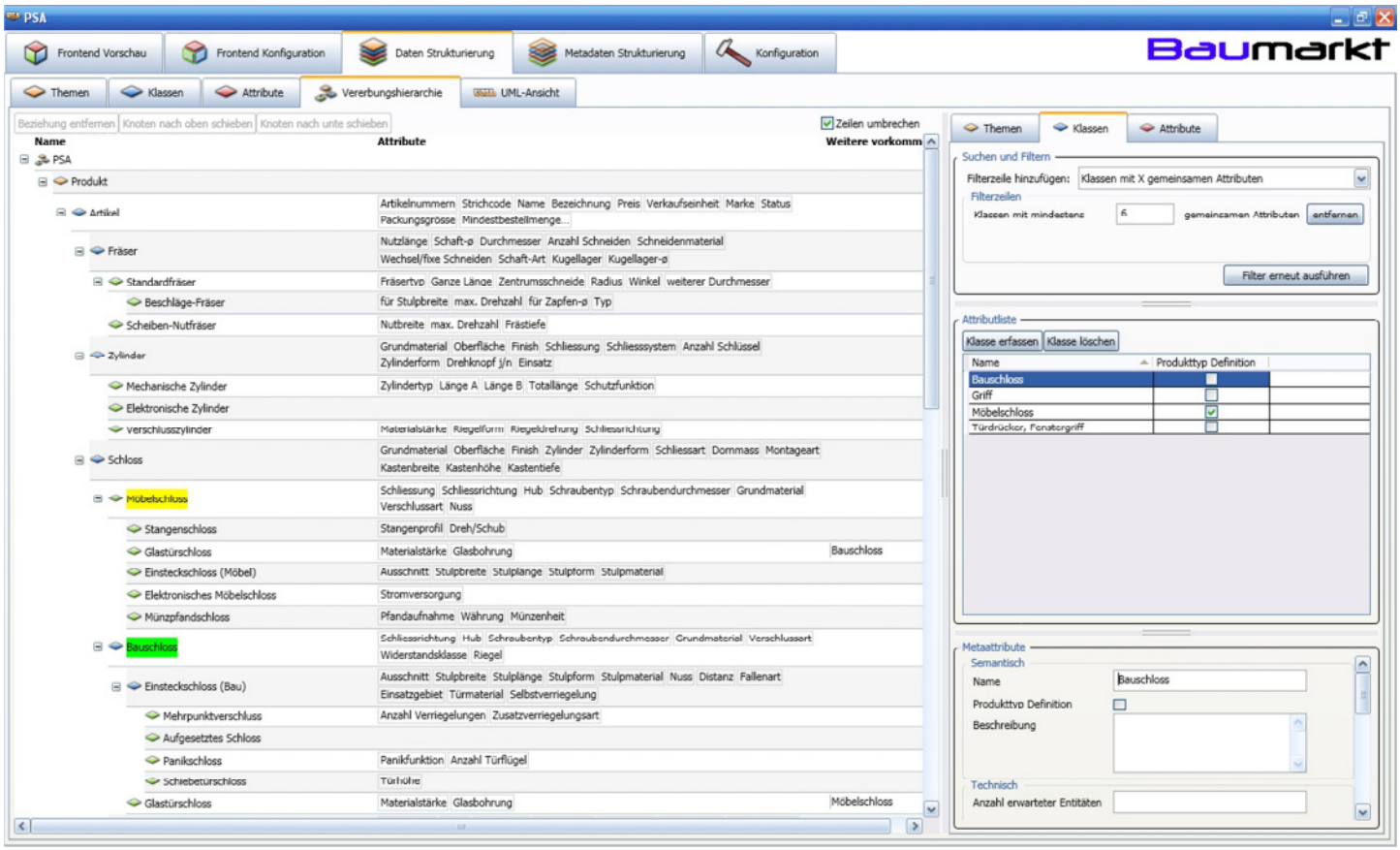


Abbildung 5: Screen Vererbungshierarchie

Zentrale Ansicht Der Tab Vererbungshierarchie ist der zentrale Tab von PSA. Auf ihm werden die Vererbungen modelliert und die Attribute den Klassen zugewiesen. Mit Filtern und Markierungen lässt sich die Produktstruktur erforschen.

Vererbungsbaum Der Vererbungsbaum stellt die ganze Vererbungshierarchie platzsparend und übersichtlich dar. In der ersten Spalte sind die Klassen mit ihren Kindern zu sehen, in der zweiten Spalte die zugehörigen Attribute. Die Spalte „Weitere Vorkommen“ zeigt an, welche anderen Eltern eine Klasse auch noch hat.

Themen Um die Vererbungshierarchie zu strukturieren werden verschiedene Aspekte in verschiedenen Themen modelliert. Typische Themen sind „Produkt“, „Community Content“, „Statistik & Marketing“, „ERP“, etc. Themen haben keinen eigentlichen Einfluss auf die Vererbungshierarchie, sie bieten nur das Gefäss um verschiedene Aspekte gekapselt zu behandeln.

Filter Die Filter beziehen sich in erster Linie auf die Liste um die angezeigten Klassen, bzw. je nach Tab auch Attribute oder Themen, einzuschränken. Es können verschiedene Filter gleichzeitig gesetzt sein, diese werden mit einem logischen „and“ verknüpft.

Markieren Einige Filter bieten noch zusätzlich Markierungen: Der angewendete Filter

„Klassen mit X gemeinsamen Attributen“ markiert die in der Liste ausgewählte Klasse grün und alle anderen Klassen, welche mindestens sechs Attribute gemeinsam haben gelb.

Generische Filter

Die meisten Filter beziehen sich auf die Metaattribute und arbeiten mit Kriterien wie „ist ausgefüllt“ oder „beinhaltet“. Diese Filter sind generisch und somit auch auf selbst erstellten Metaattributen anwendbar. Sie sind sowohl für Attribute, Klassen als auch Themen anwendbar.

Ähnliche Namen

Alle Baumelemente lassen sich nach ähnlichen Namen filtern. Wird z.B. nach dem Filtern von Themen mit ähnlichen Namen ein Thema in der Liste ausgewählt, wird dieses im Baum grün und alle Themen mit ähnlichen Namen gelb markiert.

Filter für Klassen

Folgende Filter stehen für Klassen zur Verfügung:

- Klassen mit X gemeinsamen Attributen: Wie im Screenshot gezeigt, markiert er Klassen, welche mindestens X gemeinsame Attribute haben. Dies kann ein Hinweis darauf sein, dass eine gemeinsame Vaterklasse möglich wäre.
- Nicht im Vererbungsbaum (und nicht erbend): Mit diesem Filter lassen sich Klassen finden, welche noch nicht im Vererbungsbaum sind. Wird eine Elternklasse aus dem Vererbungsbaum entfernt, sind auch die Kinder nicht mehr sichtbar. In einem solchen Fall zeigt dieser Filter nur die Elternklasse an.
- Nicht-Produkttypdefinitionen ohne Kinder: Dieser Filter gibt alle abstrakten Klassen an, von denen auch indirekt keine Produkttypdefinitionen, also nicht abstrakten Klassen, erben.

Filter für Attribute

Für Attribute stehen folgende spezialisierten Filter zur Verfügung:

- Mehrfach verwendete Attribute: Bei diesem Filter lässt sich die Mindestanzahl Vorkommen definieren. Wird ein Attribut selektiert werden im Baum alle Vorkommen grün markiert. Dies geschieht unabhängig davon, ob dieser Filter gesetzt ist oder nicht.
- Nie verwendete Attribute: Dieser Filter zeigt erfasste aber nicht verwendete Attribute an. Dies kann z.B. nach einem Import von Attributlisten sehr nützlich sein.

Modellierung

Auf dieser Ansicht geschieht die Modellierung hauptsächlich durch Drag and Drop. So können Klassen sowohl von der Liste rechts in den Baum eingehängt als auch innerhalb des Baumes verschoben werden. Auch die Attribute lassen sich von der Liste zu den Klassen ziehen, zwischen den Klassen verschieben und innerhalb einer Klasse sortieren.

B.2.2 UML Ansicht

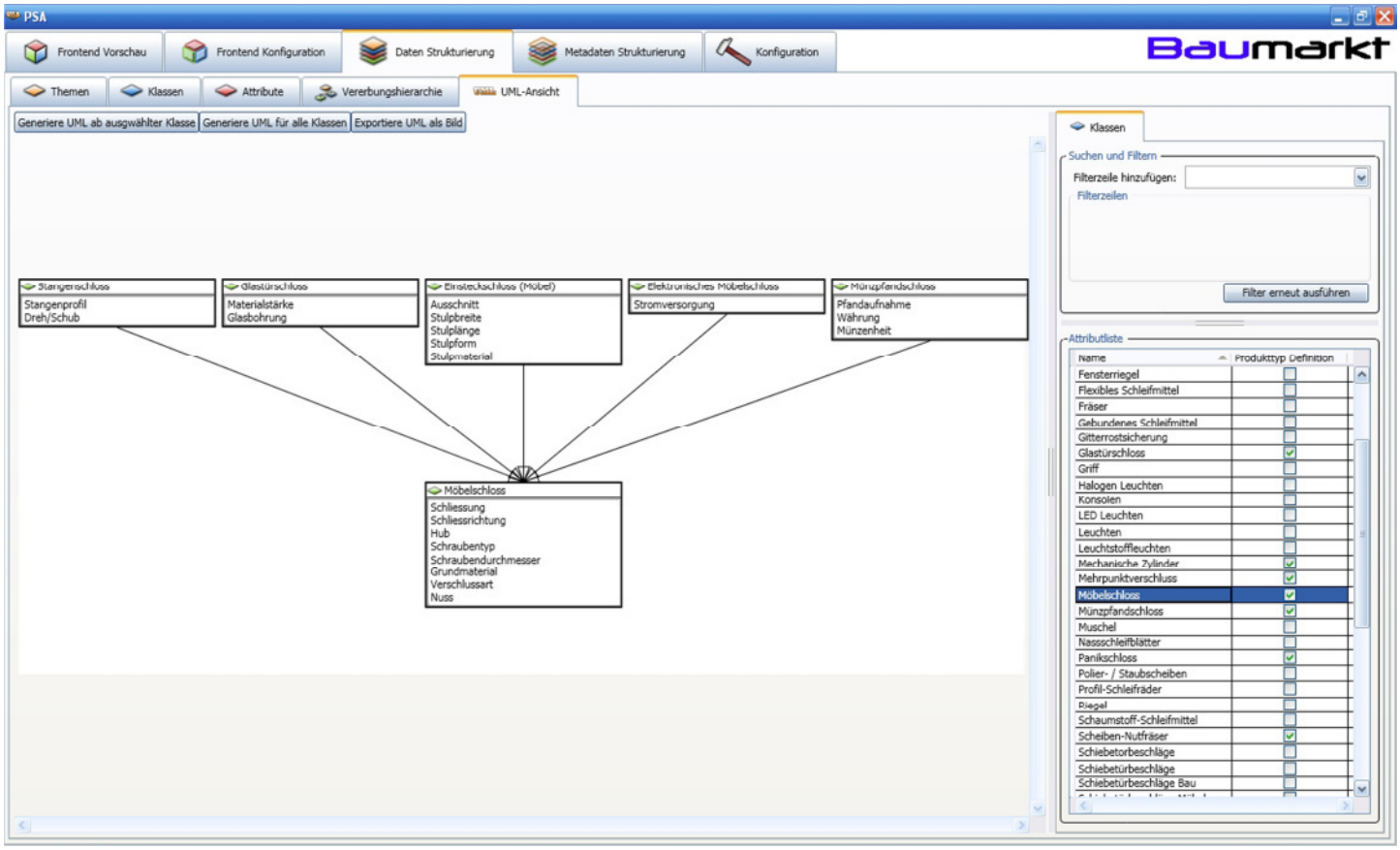


Abbildung 6: Screen UML Ansicht

UML Klassendiagram

Auf dieser Ansicht lässt sich ein UML Klassendiagramm generieren. Es kann gewählt werden von welcher Elternklasse aus das UML generiert wird. Im Beispiel wurde die Klasse Möbelschlösser gewählt. Bei Bedarf können die Klassen auch von Hand neu angeordnet werden.

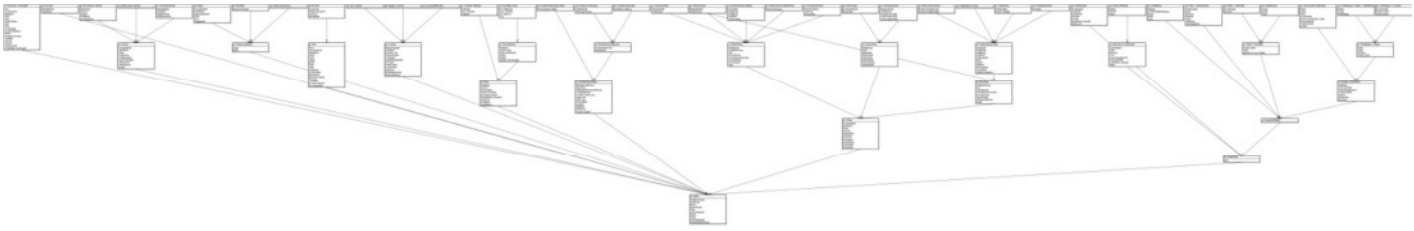


Abbildung 7: UML Klassendiagramm aller Beispieldaten

UML braucht zu viel Platz

Wir haben ein UML Diagramm von all unseren Beispieldaten generieren lassen. Obwohl in diesem Beispiel nur ein sehr kleiner Teil einer ganzen Produktstruktur modelliert wurde, ist das so generierte Diagramm kaum mehr zu gebrauchen.

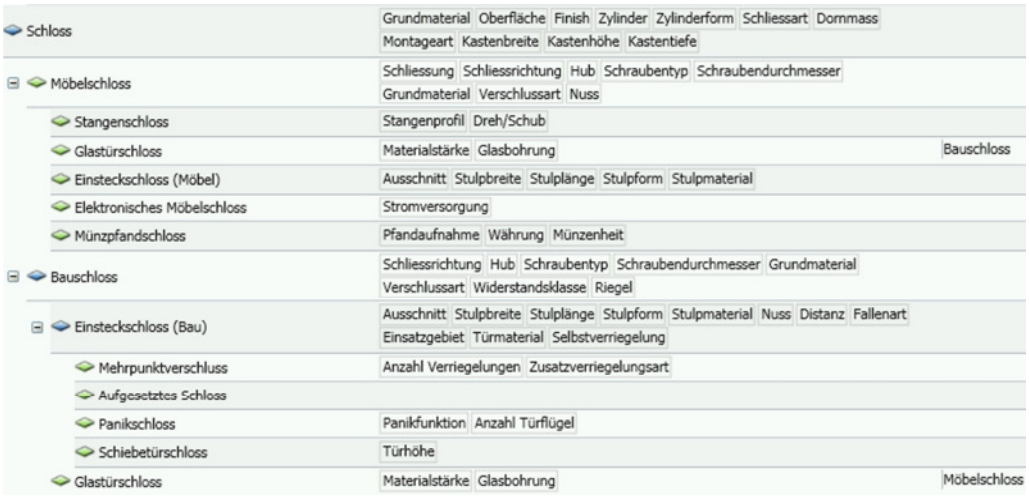


Abbildung 8: Vererbungshierarchie der Schlösser als Baum

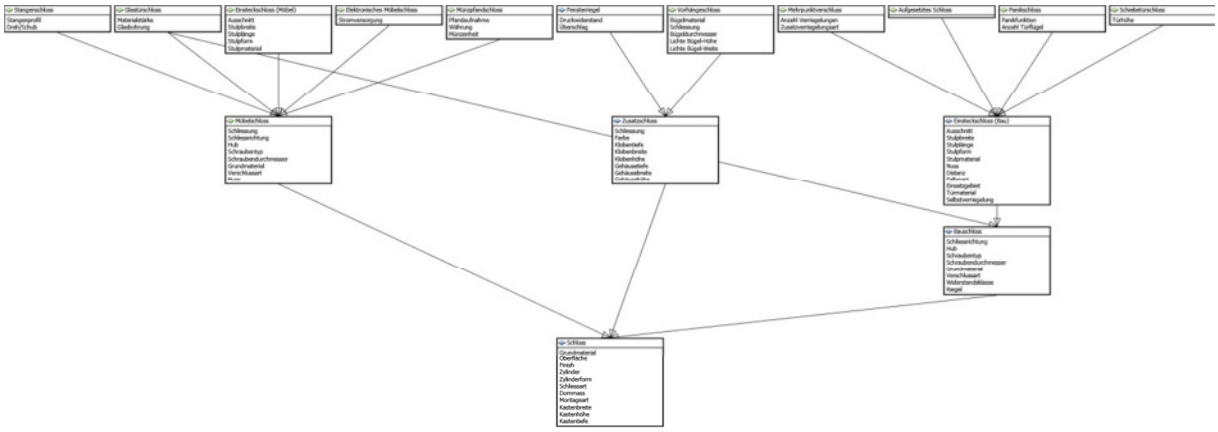


Abbildung 9: Vererbungshierarchie der Schlösser als UML

Vergleich

Ein Vergleich derselben Daten in der Baumansicht und als UML zeigt wie kompakt und platzsparend die Baumansicht ist.

UML Editor

Unser erster Ansatz für die Modellierung der Vererbungshierarchie war ein UML Klassendiagramm Editor. Wie zu sehen ist kann ein umfangreiches UML nicht komplett auf einem Bildschirm dargestellt werden. Ein Editor welcher immer nur einen kleinen Ausschnitt der zu modellierenden Daten anzeigen kann, halten wir aber für ungenügend. Mit der Baumdarstellung haben wir eine Lösung gefunden, bei welcher alle Zusammenhänge der aktuell zu modellierenden Klassen angezeigt werden können.

B.2.3 Listenansicht

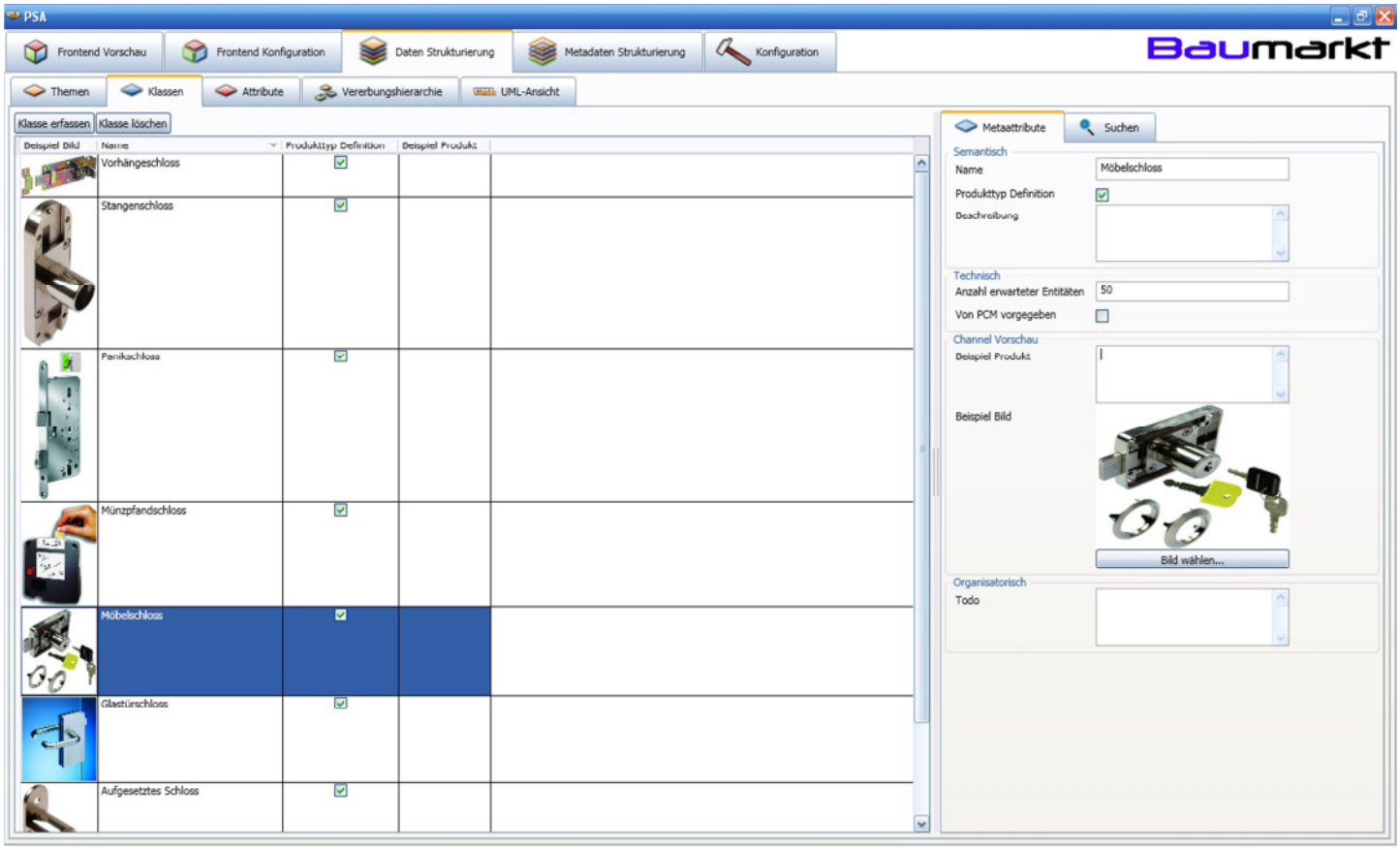


Abbildung 10: Listenansicht der Klassen

Listenansicht

Attribute, Klassen und Themen lassen sich in Listenansichten darstellen. Unter Suche stehen dieselben Filter zu Verfügung wie auf dem Vererbungshierarchie Tab. Natürlich lässt sich die Liste auch sortieren. Die in der Liste angezeigten Metaattribute sind frei wählbar.

Metaattribute

PSA bietet eine Auswahl von Metaattributen an. Bei Bedarf können aber weitere Metaattribute selbst definiert oder auch bestehende gelöscht werden.

Klassen Metaattribute

Bis auf die Produkttyp Definition brauchen die Metaattribute keine Erklärung. In der Informatik werden abstrakte Klassen und instanzierbare Klassen unterschieden. Genau dieselbe Idee steckt hinter der Produkttyp Definition: Mit diesem Metaattribut werden Klassen gekennzeichnet, welche einen Artikel vollständig beschreiben. Das heisst im produktiven Einsatz werden alle Artikel einer Produkttyp Definition zugewiesen und mit den dadurch bestimmten Attributen im PIM erfasst.

Themen Metaattribute

Die Themen sind ein strukturierendes Element, haben aber keinen weiteren Einfluss auf die Produktstruktur. Trozdem sind in der Standard Konfiguration nebenstehende Metaattribute möglich.

Abbildung 11: Themen Metaattribute

Attribut Metaattribute

Bei den Attributen können viele Informationen der Produktstruktur in Metaattributen festgehalten werden. Es zeigt sich aber auch ein Problem: Wie viele Metaattribute sind sinnvoll? Bei hunderten von Attributen ist es sehr arbeitsintensiv so viele Metadaten zu erfassen. Es muss also von Projekt zu Projekt entschieden werden, welche Metaattribute relevant sind.

Eine wichtige Diskussion ist bei jeder Produktstrukturierung, welche Attribute Variationen unterscheiden. Dies ist keine technische Fragestellung, sondern eine Frage des Marketings. Es gibt Attribute welche nur bei gewissen Produkttypen als Unterscheidung von Variationen dienen. Zur Diskussion stand, ob man diese Information für jede Zuordnung eines Attributs zu einer Klasse separat erfassen will.

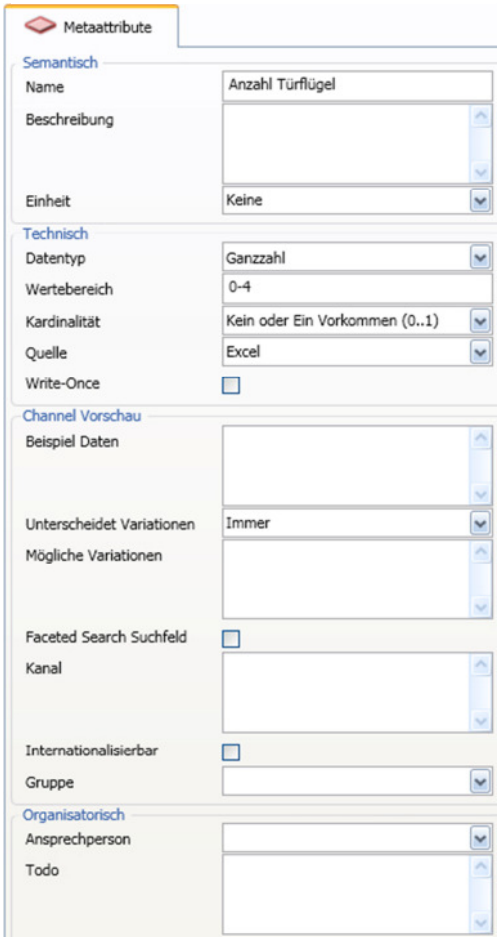


Abbildung 12: Attribut Metaattribute

Mit Metaattributen welche sich auf die einzelnen Zuordnungen von Attributen zu Klassen beziehen, wäre der Aufwand für deren Verwaltung massiv gestiegen. Daher bietet PSA diese Möglichkeit nicht. Um die Resultate der Diskussion um die Variationen sinnvoll zu erfassen, bieten wir zwei Metaattribute an:

- „Unterscheidet Variationen“ kann die Werte „Nie“, „Manchmal“ und „Immer“ annehmen.
- Im Feld „Mögliche Variationen“ kann weiter spezifiziert werden, in welchen Fällen dieses Attribut nun Variationen unterscheidet, und in welchen nicht.

B.2.4 Eingabe Masken

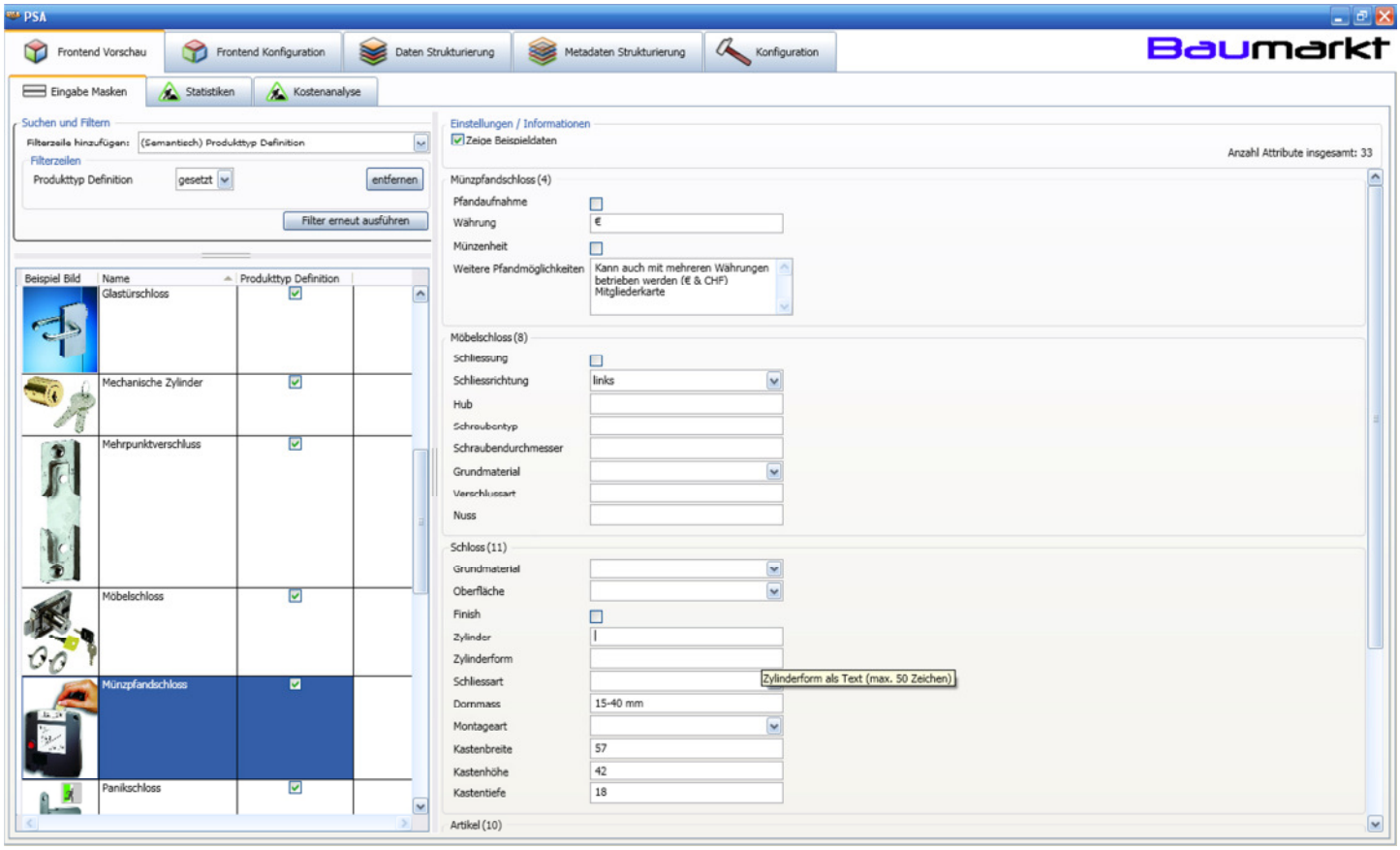


Abbildung 13: Screenshot Eingabe Masken

Produkt orientiert

Die Eingabe Masken sind eine weitere zentrale Ansicht von PSA. In dieser Ansicht sieht man aus Sicht einer Klasse, typischerweise einer Produkttyp Definition, das Resultat der Produktstrukturierung. Es wird eine Eingabemaske generiert, welche alle zu erfassenden Attribute aufzeigt. Aufgrund des Datentyps des Attributs werden die Eingabefelder passend formatiert. Der Tooltip entspricht dem Metaattribut Beschreibung. Auf Wunsch können die Beispieldaten eingeblendet werden.

Diskussions-grundlage

Diese Ansicht bietet eine wichtige Diskussionsgrundlage. In einer auch für User vertrauten Darstellungsweise kann die Modellierung eines Produkttyps schnell begriffen werden. Es ist eine sehr konkrete Darstellung des Resultats, sie unterstützt damit die Erkennung allfälliger Mängel, z.B. das Fehlen von Attributen. Man kann sich aber auch sehr gut vorstellen, wie aufwändig die Erfassung und Pflege von Artikeln sein wird. So können auch überflüssige Attribute identifiziert werden.

B.2.5 Enumerationen

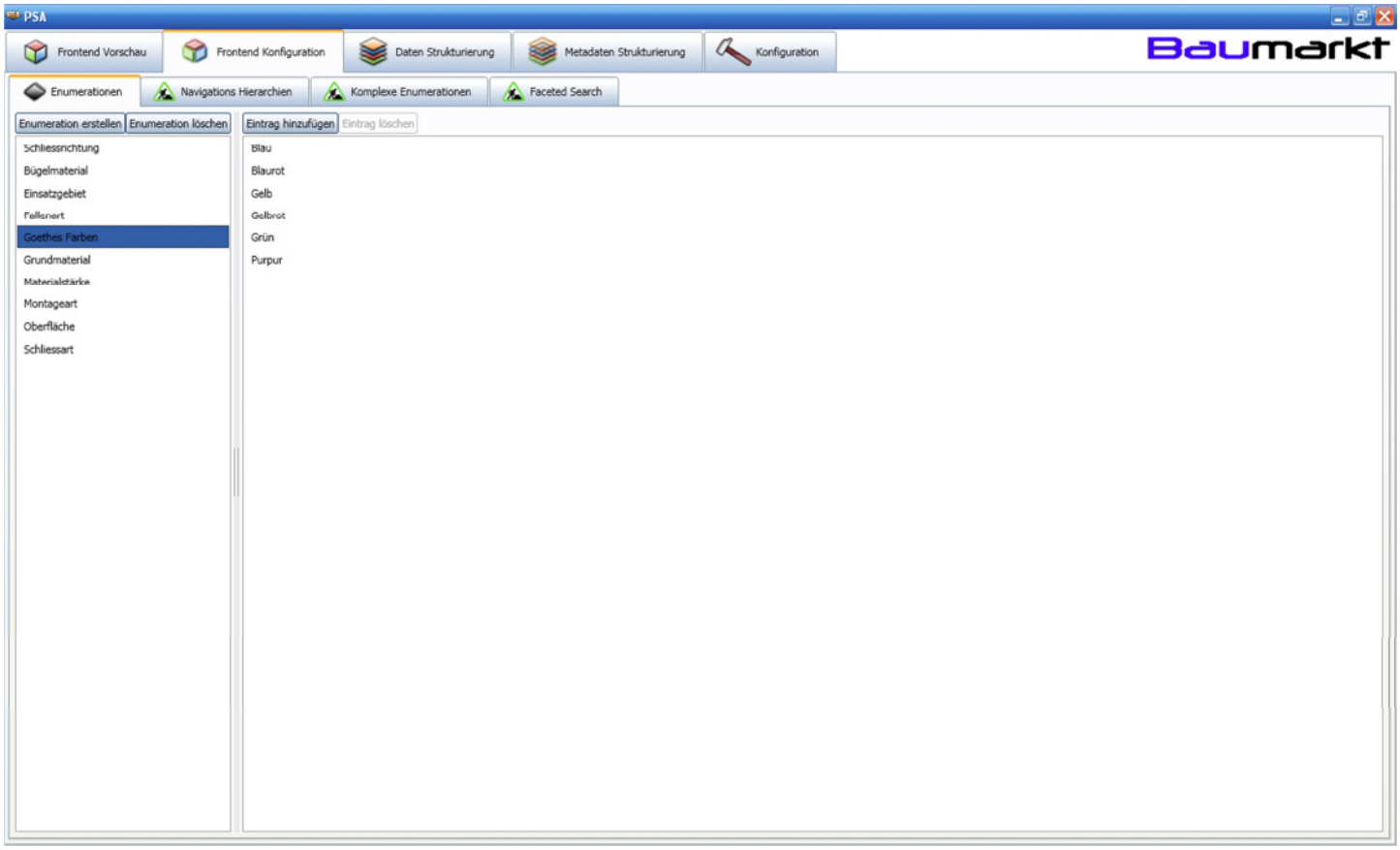


Abbildung 14:Screen Enumerationen

Produktiv Daten

Ein Diskussionspunkt während der Produktstrukturierung sind Enumerationen. So gibt es z.B. verschiedene Farbräume, bei jedem Kunden muss abgeklärt werden, mit welchen Farbbezeichnungen er arbeiten möchte. Die in diesem Tab erfassten Enumerationen werden später ins Produktivsystem übernommen.

Metadaten

Auch im Haupttab Metadaten Strukturierung können Enumerationen modelliert werden; beide Enumrations Tab haben denselben Aufbau. Die dort modellierten Enumerationen dienen jedoch als Basis für die Metaattribute.

B.2.6 Metadatenstrukturierung

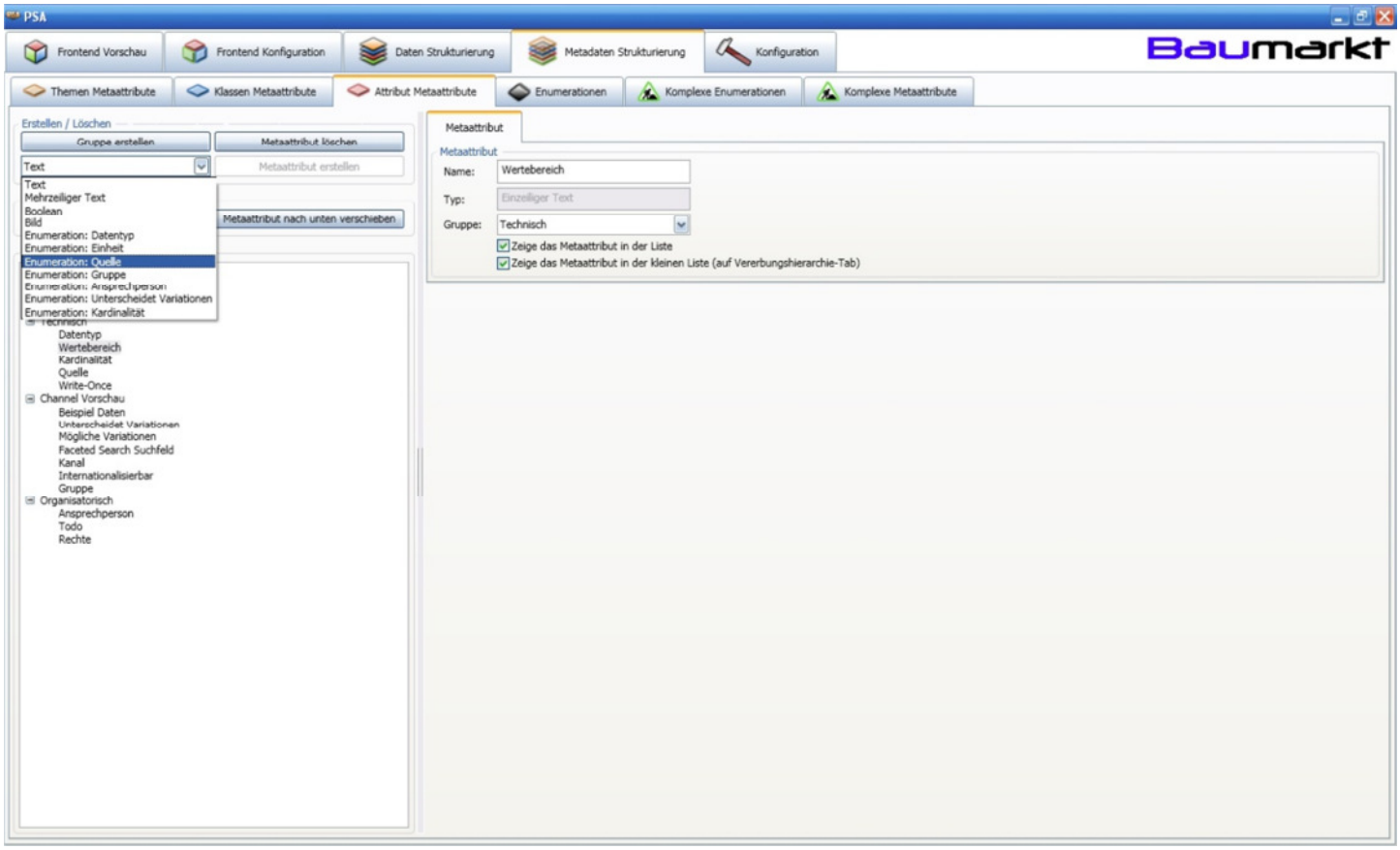


Abbildung 15: Screen Metadatenstrukturierung

Metaattribute Die Metaattribute können für jedes Projekt individuell angepasst werden. Sie können sortiert und in Gruppen organisiert werden. Für ein Metaattribut muss jeweils dessen Datentyp festgelegt werden. Zur Auswahl stehen nebst Text, Boolean und Bild auch Enumerationen.

B.2.7 Konfiguration

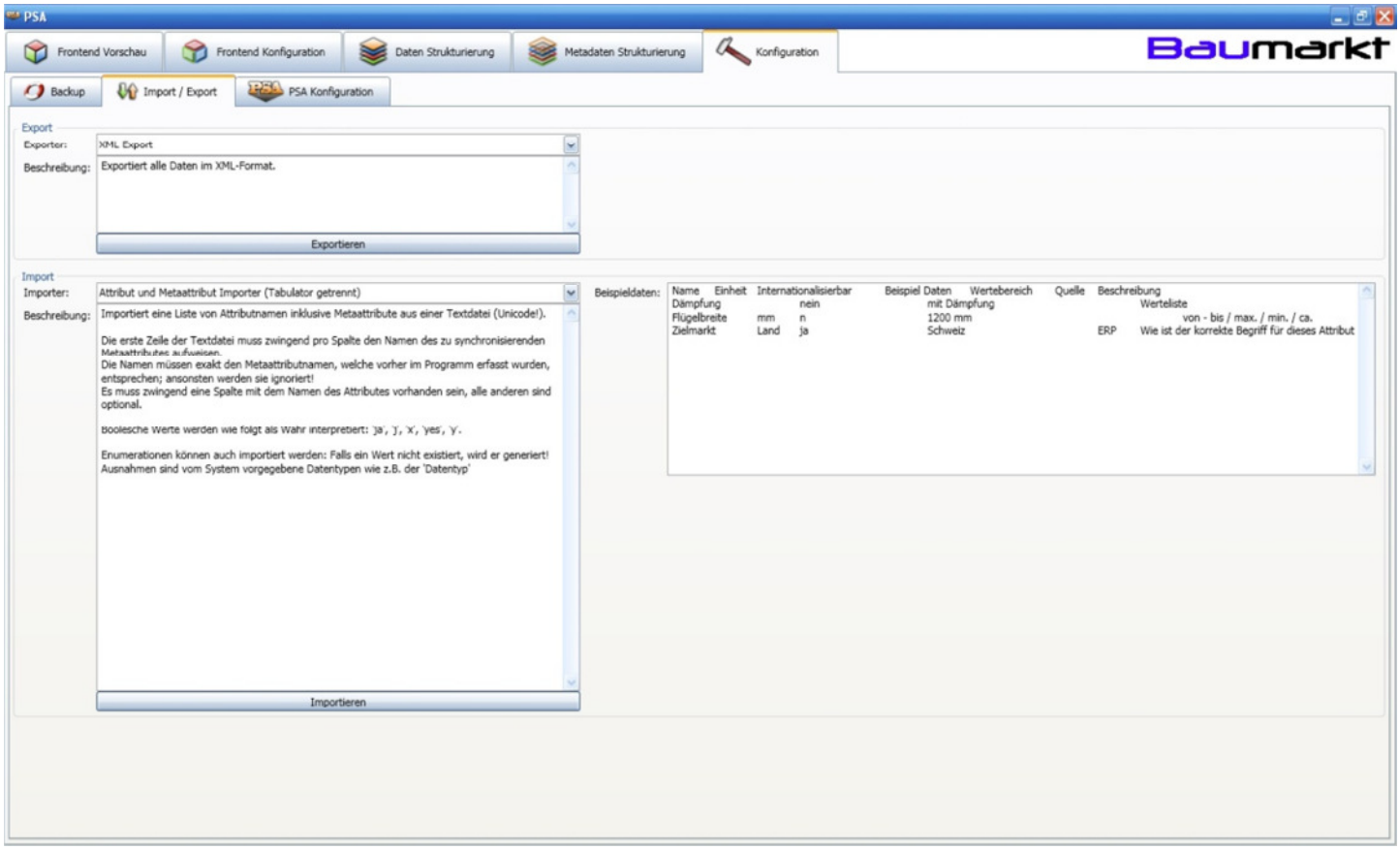


Abbildung 16: Screen Import / Export

XML Export Mit dem XML Export können sämtliche in PSA modellierten Daten exportiert werden.

Import PSA unterstützt verschiedene Formate für den Import:

- Eine einfache Liste von Attributen
- Attribute mit Metaattributen
- Vererbungshierarchie, inkl. Attributen

Alle diese Formate können einfach in Excel erstellt werden.

Backup Es lässt sich ein Backup mit allen Daten (inkl. Beispielbilder) erstellen und wieder einlesen.

Konfiguration Auf dem Konfigurations Tab kann das Kundenlogo gesetzt werden.

B.3 Nicht umgesetzte Ideen

B.3.1 Modellierung via Präsentationsmatrix

Idee

Es geht um einen komplett anderen Ansatz die Produkttypen zu modellieren. Die Idee ist, dass man nicht an einer abstrakten Modellierung arbeiten muss: Wir wollen gleich die verschiedenen Darstellungsmöglichkeiten erarbeiten. Schlussendlich ist das Ziel eines PIM, dass die Produktdaten auf verschiedenen Kanälen optimal präsentiert werden. Diese Präsentationen würden gleich mit modelliert.

Präsentations-matrix

Die Präsentationsmatrix ist eine mehrdimensionale Matrix. Folgende Dimensionen hätten wir uns vorgestellt:

- Produkttyp
- Kanal (Mobile, Webshop, Katalog, Programm für Logistiker, ...)
- Darstellungsform (kurze Darstellung (z.B. Suchergebnis), mittlere Darstellung (z.B. Auflistung aller Artikel), lange Darstellung (z.B. Detailseite)
- Marketingkampagne
- Saison

Natürlich muss man auch weitere Dimensionen erfassen können um die projektspezifischen Bedürfnisse abdecken zu können.

Jede Zelle dieser Matrix repräsentiert eine konkrete Darstellung eines Produkttyps; somit ist dies ein Knoten der Customer-Journey. Es ist auch möglich, dass es nicht alle möglichen Darstellungsformen gibt, d.h. dass gewisse Zellen der Matrix nicht definiert werden müssen; falls z.B. nicht alle Produkttypen im Katalog angeboten werden, bleiben entsprechende Felder leer.

Workflow

Der Workflow sieht folgendermassen aus (wobei natürlich ein iteratives Vorgehen möglich ist):

- Auflisten der Attribute (wie bisher)
- Auflisten der Produkttypdefinitionen (ohne Attribute, nur Namen)
- Erstellen der Präsentationsmatrix
- Zuweisen der Attribute zu den Zellen der Präsentationsmatrix (d.h. bestimmen, welche Attribute für welchen Produkttyp auf welchem Kanal in welcher Darstellungsform anzuzeigen ist.)

Definition einer Zelle

Um eine Zelle zu definieren, braucht es zwei wesentliche Angaben: Welche Produkte werden zu einem Eintrag dieser Ansicht zusammengefasst und welche Attribute werden angezeigt.

Der erste Punkt behandelt die Unterscheidung von Variationen. Es wird bestimmt ob z.B. eine Blaue Levis 501 und eine Schwarze Levis 501 als eigenständige Produkte angezeigt werden oder ob es nur einen Eintrag für beide Farben gibt.

Redundante Arbeit vermeiden

Es ist davon auszugehen, dass viele dieser Zellen in ein logisches Muster passen. So sind wahrscheinlich alle Attribute einer kurzen Ansicht auch in der mittleren und langen Ansicht zu sehen. Auch werden viele Produkttypen in gewissen Darstellungen exakt dieselben Attribute benötigen.

Dies kann man ausnützen um den Arbeitsaufwand zu minimieren: Zwischen den Zellen kann man verschiedene Abhängigkeiten definieren. So kann man z.B. die Dimension kurz - mittel - lang als Hierarchie definieren: Somit werden automatisch alle Attribute der kurzen Darstellung auch der mittleren und langen hinzugefügt.

Generieren der Produkttyp-definitionen

Aus den so gemachten Angaben hätten sowohl die Produkttypdefinitionen als auch die Vererbungshierarchie generiert werden können. Anhand dieser hätte man einen Überblick über die gemachte Modellierung gewinnen können: Man hätte eine Übersicht über alle Attribute gehabt, welche einen Produkttyp definieren.

Feedback von Herrn Keller

- Eine solche Matrix hat zu viele Felder: $O(n^{\text{Anzahl Dimensionen}})$. Es wäre zu viel Arbeit, dies alles auszufüllen. In der Praxis wird das vereinfacht: Z.B. gibt es für alle Artikel nur eine Darstellungsform "Kurz". In dieser dürfen halt nur Attribute vorkommen, die für alle Artikel definiert sind. Um trotzdem auf die Eigenheiten der Artikel eingehen zu können, gibt es ein Attribut "kurzer Beschreibungs-Text".
- Die Verwaltung, welche Informationen über welche Kanäle dargestellt werden, ist nicht Aufgabe des PIM.

B.3.2 Navigationshierarchie

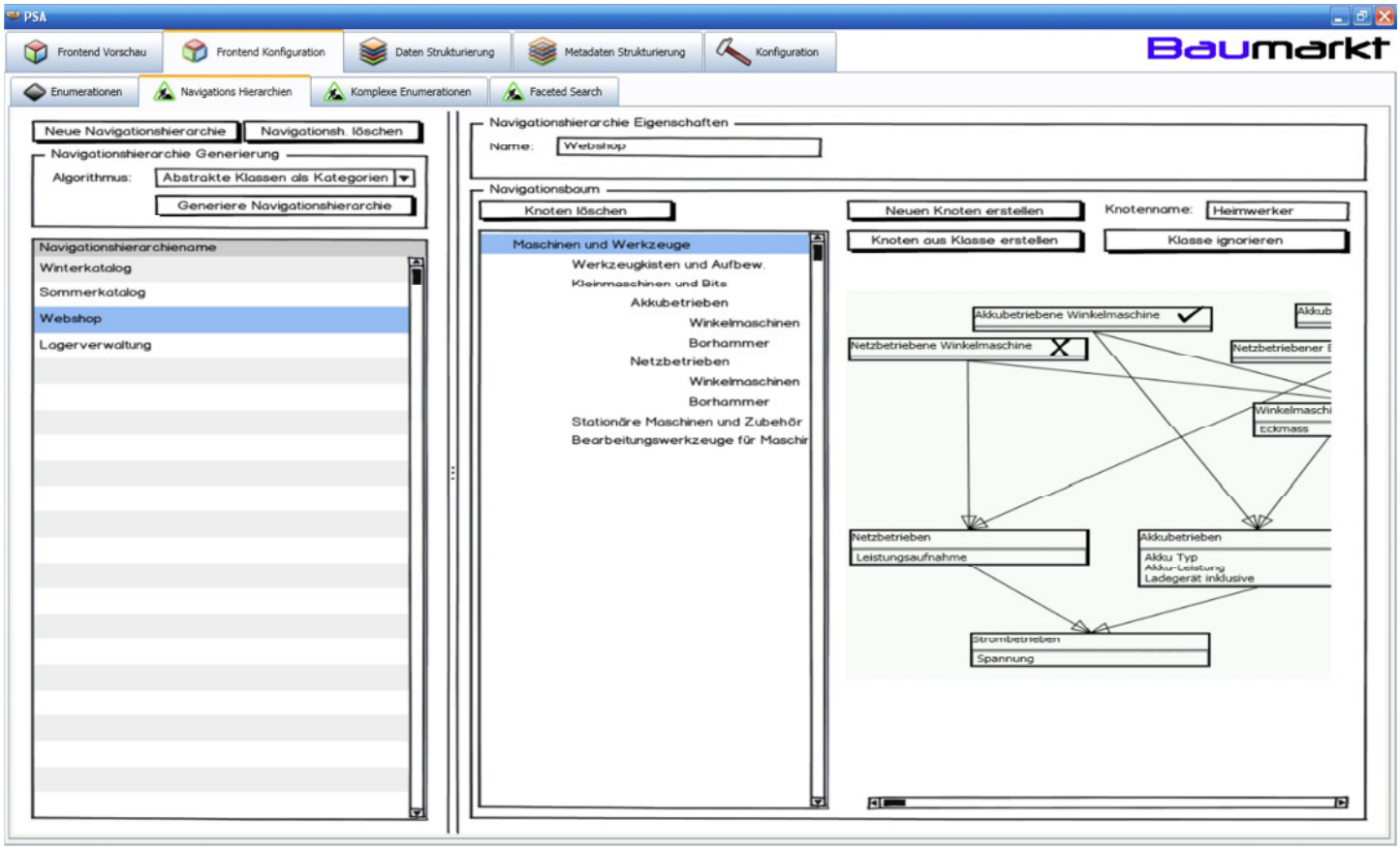


Abbildung 17: Wireframe Navigationshierarchie

Navigations-hierarchien

Damit der Kunde in einem umfangreichen Sortiment sein gewünschtes Produkt findet, müssen die Produkte in einer sauberen und intuitiv verständlichen Struktur geordnet sein. Je nach Kanal, manchmal sogar je nach Saison, gibt es verschiedene Anforderungen an die Navigationshierarchie. Querverweise auf Zubehör und andere Zusammenhänge zwischen Produkten machen die Modellierung richtig anspruchsvoll.

Logischer Aufbau

Mit hoher Wahrscheinlichkeit haben die Navigationshierarchien und die Vererbungshierarchie einen ähnlichen Aufbau. Unsere Idee war, dass man in einem ersten Schritt eine Navigationshierarchie aus der Vererbungshierarchie generiert. Anschliessend hätte man das Resultat den konkreten Bedürfnissen anpassen können.

B.3.3 Komplexe Enumerationen

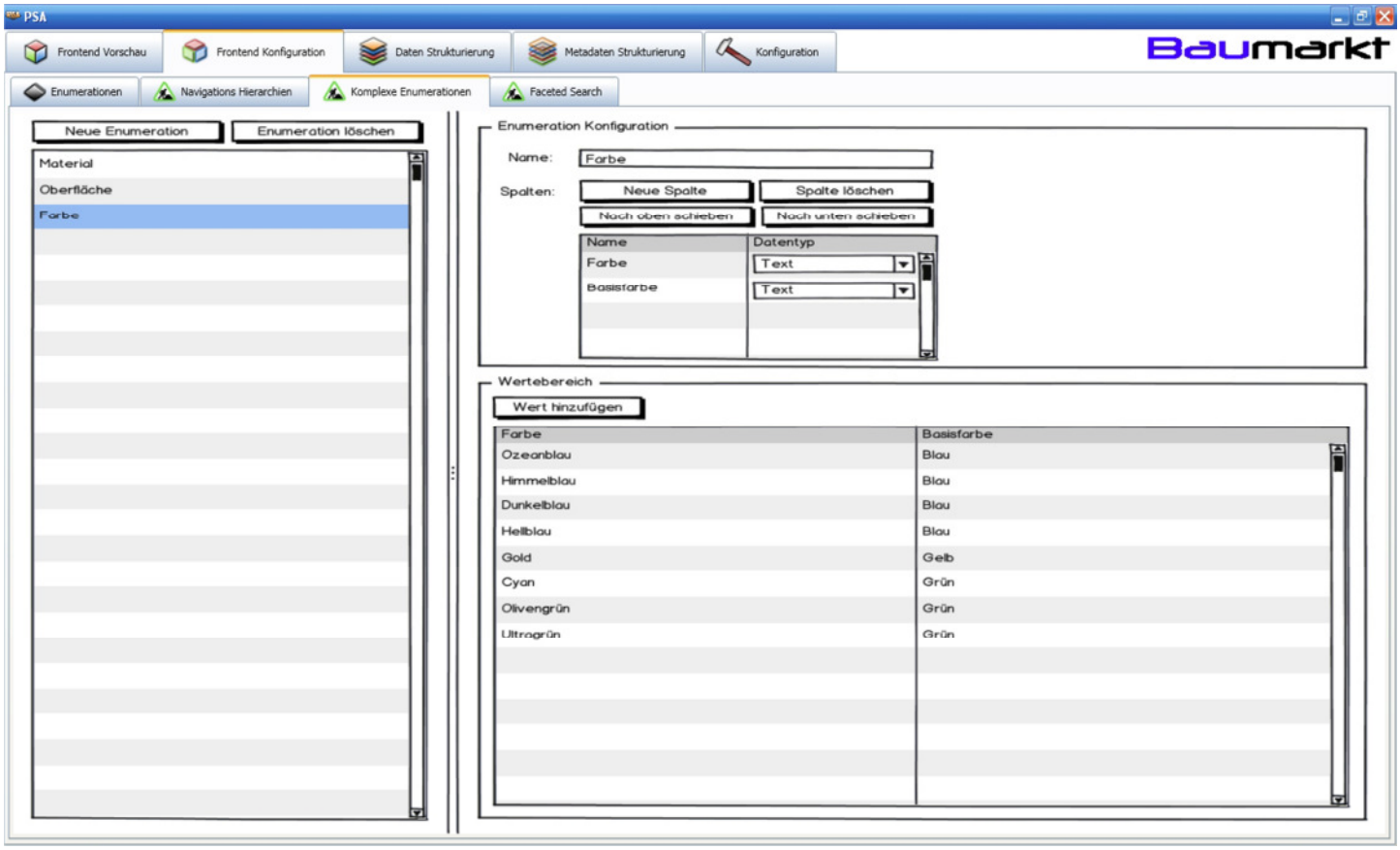


Abbildung 18: Wireframe Komplexe Enumerationen

Mehrere Felder

Unter dem Arbeitstitel Komplexe Enumerationen wollten wir Enumerationen mit mehreren Spalten anbieten. Es wären verschiedene Datentypen für die Spalten unterstützt worden; insbesondere hätten andere Enumerationen als Basis für eine Spalte dienen können.

Farben

Als Beispiel sollen auch hier die Farben dienen. Es wäre möglich gewesen, den Vermarktungsfarben eine Grundfarbe zuzuordnen. Im produktiven Einsatz könne solche Zuordnungen wichtig werden bei der Suche nach Produkten welche zusammen passen.

Datentyp

Auch für die Metadatenstrukturierung wären die komplexen Enumerationen nützlich gewesen. Um in der Maskenansicht passende Eingabefelder anzuzeigen, ist PSA darauf angewiesen, dass das Metaattribut Datentyp sinnvoll ausgefüllt ist. In der heutigen Implementatin können nur von uns vorgegebene Datentypen ausgewählt werden, da neue Einträge von der Maske nicht erkannt würden. Mit zweispaltigen Metaattributen hätte die erste Spalte frei modelliert werden können, während in der zweiten Spalte das Anzeigeformat für die Eingabemaske hätte ausgewählt werden müssen.

B.3.4 Komplexe Metaattribute

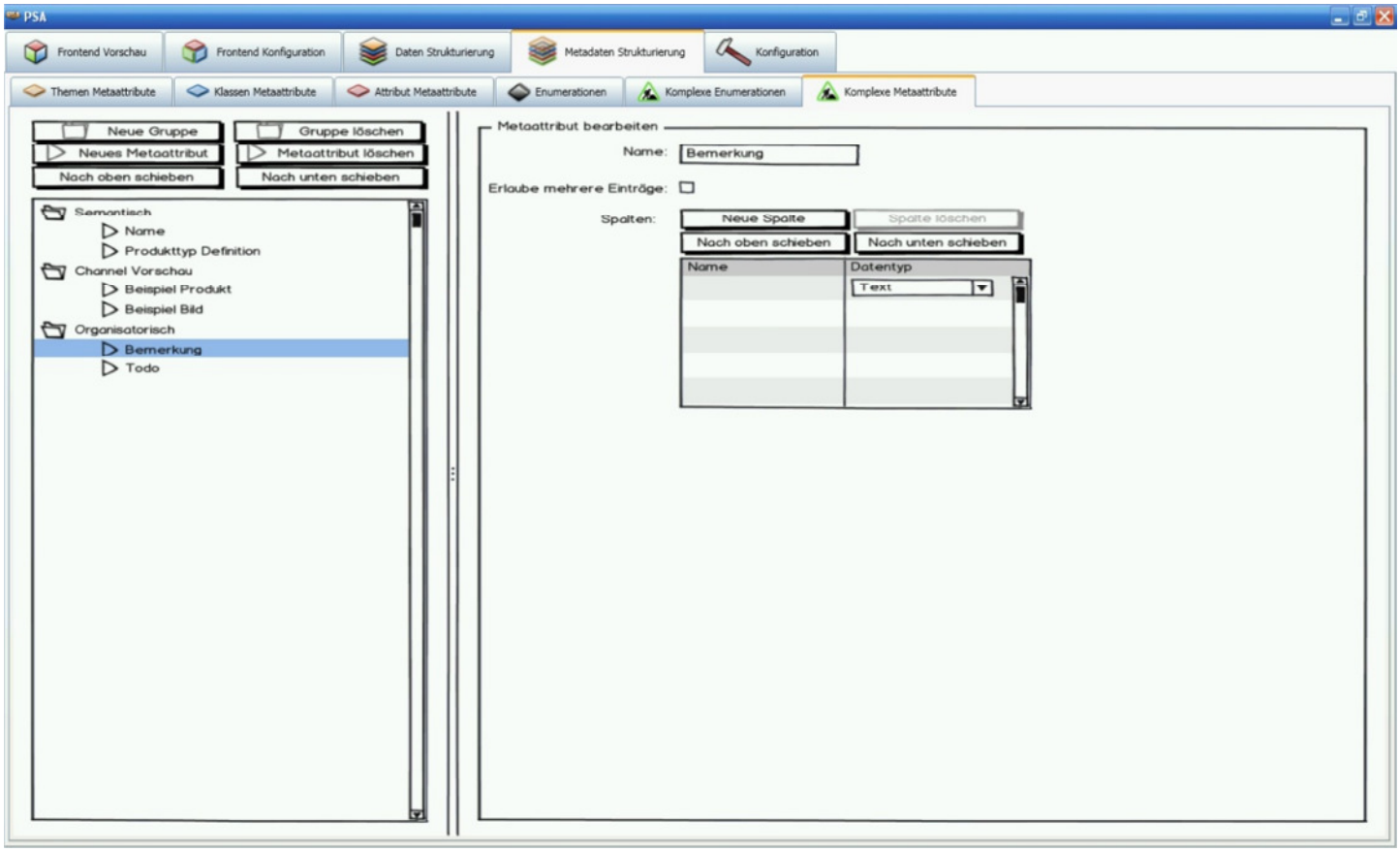


Abbildung 19: Wireframe Komplexe Metaattribute

4 Typen

Wir wollten vier Typen von Metaattributen anbieten:

- Metaattribute welche genau einen Wert aufnehmen. Diese sind heute implementiert.
- Metaattribute welche mehrere Werte aufnehmen. Beispielsweise hätten so mehrere Todos in separaten Feldern erfasst werden können.
- Metaattribute mit mehreren Spalten. Für die Ansprechperson wäre vorgesehen gewesen, dass nebst dem Namen auch eine Telefonnummer hätte eingegeben werden können.
- Tabellen: Für die Modellierung von Zugriffsrechten wären die Spalten Rolle, Leserecht und Schreibrecht nötig. Zusätzlich müssten mehrere solcher Einträge möglich sein.

B.4 Software Architektur

B.4.1 Layer

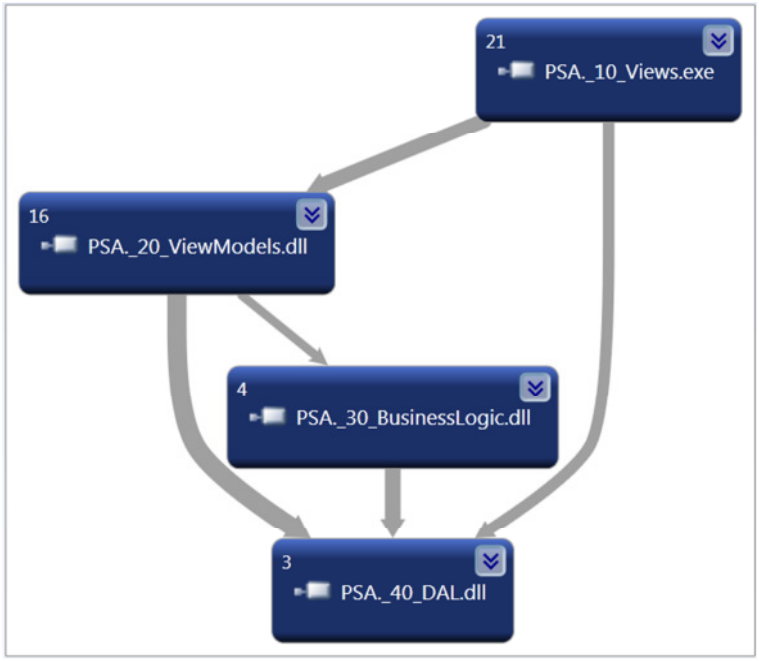


Abbildung 20: Übersicht über die Layer

Views

In .XAML Files ist das Aussehen des GUI beschrieben. In einigen Fällen mussten wir das Verhalten der Ansichten durch Code Behind weiter verfeinern.

ViewModels

Die ViewModel stellen den Views die benötigten Daten zur Verfügung und verwalten deren Zustand. Dies entspricht dem MVVM Pattern. Wir haben einen pragmatischen Weg zur Umsetzung von MVVM gewählt: Wie bereits bei den View beschrieben, haben wir nicht strikt auf Code Behind verzichtet.

BusinessLogic

Logik welche weitgehend ohne Userinput ablaufen kann ist im BusinessLogic Layer organisiert. Da PSA vor allem die Modellierung und Visualisierung unterstützt, beinhaltet dieser Layer nur wenige Klassen. Dies sind z.B. die Import und Export Algorithmen

DAL

Der Data Access Layer erfüllt zwei Aufgaben:

- Definition der Datenstruktur: Alle Datenklassen, ihre Properties und Assoziationen sind im DAL definiert. Deshalb haben auch alle anderen Layer (inkl. Views) Referenzen auf den DAL.
- Zugriff auf die Daten: Dies geht vom einfachen Auslesen aller Objekte eines Typs, z.B. aller AIHAttribut Objekte, bis hin zu den komplizierteren Filtern.

B.4.2 Konzeptionelles Datenmodell

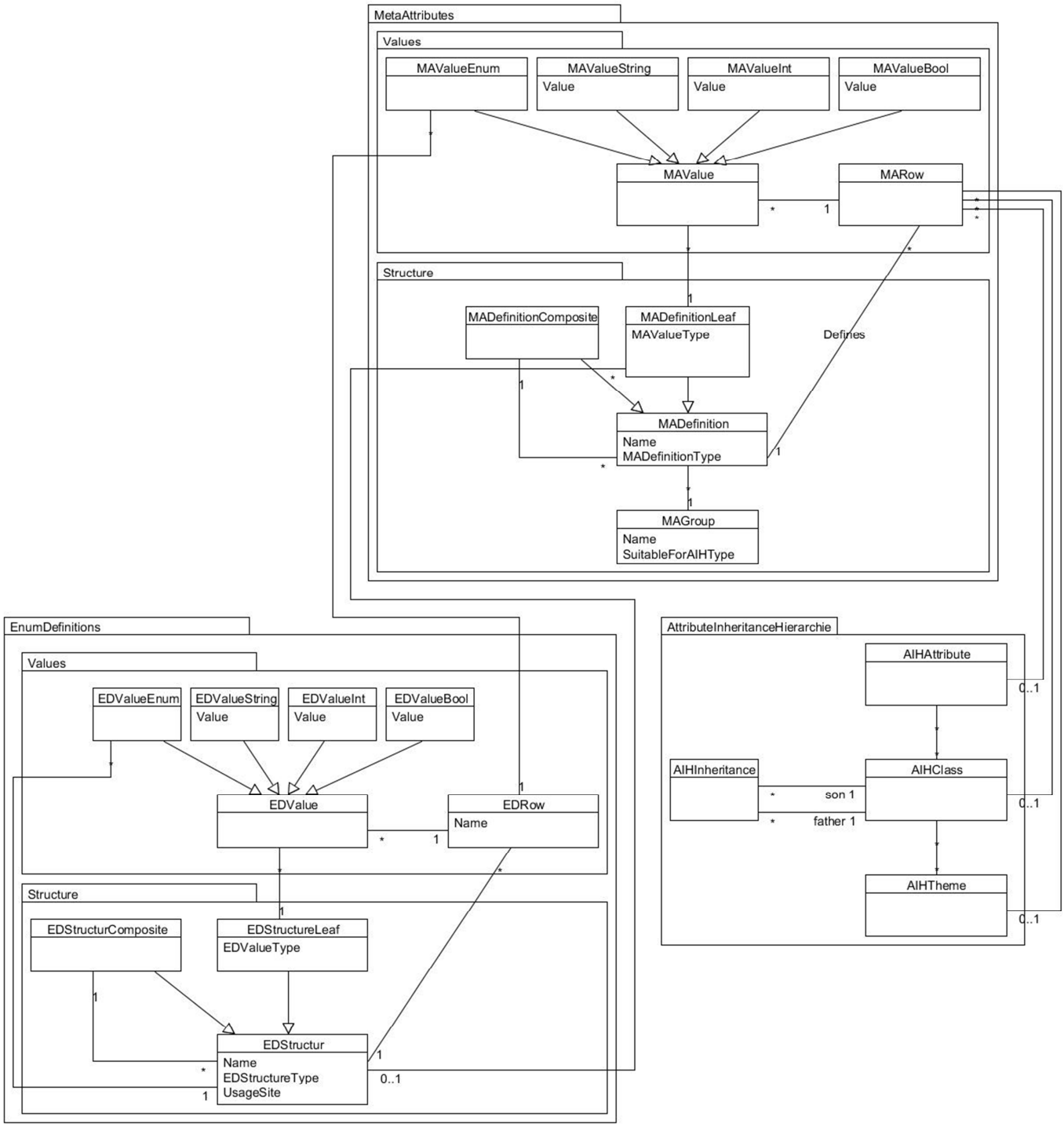


Abbildung 21: Konzeptionelles Datenmodell

B.4.2.1 Meta Attributes

Anforderungen

Es ist nicht möglich eine abschliessende und für alle Projekte passende Liste von Metaattributen zu erstellen. Daher können die Metaattribute nicht als normale Properties der AIH-Klassen modelliert werden. Somit muss das Datenbankmodell eine erweiterbare Struktur bieten.

Structure & Values

Eine Datenbank bietet die Möglichkeit, Strukturen für die zu speichernden Daten zu verwalten und Werte innerhalb dieser Strukturen zu speichern. Genau diese Funktionalität wird hier gebraucht.

Es gibt zwei Bereiche: Structure und Values. Im Bereich Structure werden die Definitionen der Metaattribute verwaltet. Im Bereich Values werden die den AIH-Objekten zugewiesenen Werte der Metaattribute gespeichert.

Dynamischer OR-Mapper

Gerne hätten wir einen dynamisch konfigurierbaren OR-Mapper eingesetzt. Um ein neues Metaattribut zu erzeugen hätte z.B. einfach ein neuer Entitätstyp erstellt werden können.

Für die .Net Plattform konnten wir keinen solchen OR-Mapper finden. Da der Auftraggeber darauf beharrt hat, keine dynamisch typisierte Programmiersprache zu verwenden, haben wir Alternativen wie z.B. Python und SQL-Alchemy nicht weiter untersucht.

SQL

Eine weitere Möglichkeit wäre der Verzicht auf einen OR-Mapper gewesen. Mit ‚create table'-Befehlen hätte die Datenbank zur Laufzeit angepasst werden können. Der Aufwand für sämtliche Datenbankzugriffe eigene SQL-Statements zu generieren und die Daten in passende Objekte abzufüllen, schien uns allerdings grösser, als eine eigene Verwaltung der Metadatendefinitionen umzusetzen.

MAGroup

Um dem User eine bessere Übersicht über die Metaattribute zu geben, haben wir sie in Gruppen eingeteilt. Das Feld SuitableForAIHType gibt an, ob sich die Metaattribute dieser Gruppe auf AIHAttribute, AIHClass oder AIHTheme beziehen.

MADefinitionLeaf

Im Feld MAValueType der Klasse MADefinitionLeaf wird angegeben, von welchem Datentyp das Metaattribut ist. Somit wird vorgegeben, von welchem konkreten Typ die zugehörigen MAValue sind (z.B. MAValueString).

B.4.2.1.1 Komplexe Metaattribute

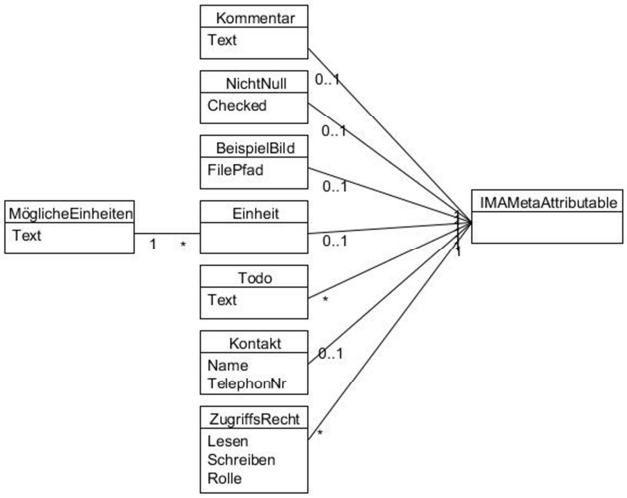


Abbildung 22: Konkretes Modell von Metaattributen

Verschiedene Typen

Wie in obiger Abbildung zu sehen ist, gibt es verschiedene Typen von Metaattributen. Es sind sowohl unterschiedliche Datentypen zu speichern, als auch verschiedene Komplexitäten zu verwalten.

Um diese Anforderung im Datenmodell umzusetzen, haben wir das Composite Pattern eingesetzt. Im Folgenden werden die 4 vorgesehenen Komplexitätstufen kurz erläutert.

Scalar

Die scalaren Metaattribute sind die Einfachsten. Für jedes IMAMetaAttributable (z.B. ein AIHClass Objekt) ist genau ein Wert zu speichern. In der Darstellung oben sind dies die Metaattribute „Kommentar“, „NichtNull“, „BeispielBild“ und „Einheit“. Die ersten drei unterscheiden sich nur im Datentyp.

Beim Metaattribut „Einheit“ gibt es eine weitere Spezialität: Die möglichen Werte sind in der Tabelle „MöglicheEinheiten“ vorgegeben. Somit verweist dieses Metaattribut auf eine Enumeration (im richtigen Datenmodell also auf ein EDStructure).

ScalarList

Für das Metaattribut „Todo“ möchten wir mehrere Einträge erlauben. Diesen Typ von Metaattribut nennen wir ScalarList.

Struct

Das Metaattribut „Kontakt“ setzt sich aus den Werten „Name“ und „Telefonnummer“ zusammen. Metaattribute mit mehreren Feldern sind Struct Metaattribute.

StructList

Die komplexesten Metaattribute speichern ganze Tabellen. So müssen für die Rechteverwaltung mehrere Einträge mit mehreren Datenfeldern gespeichert werden.

MADefinitionType

Im Feld MADefinitionType der Klasse MADefinition sind die Werte

- Scalar
- ScalarList
- Struct
- StructField
- StructList
- StructListField

möglich. Die Bezeichnungen StructField und StructListField werden für die Blätter eines Struct bzw. StructList Metaattributs verwendet.

B.4.2.2 Enum Definitions

Analog MA

Die Definitionen der Enumerationen weisen dieselbe Struktur auf wie die Definitionen der Metaattribute. Somit sind dieselben Komplexitätsstufen möglich.

UsageSite

Wir haben drei Kategorien von Enumerationen:

- Frontend: Werden vom User verwaltet. Diese Enumerationen sollen später im Produktiv eingesetzten PIM verwendet werden. Z.B. können so die verschiedenen Farbräume definiert werden.
- Metadata: Es ist möglich Metaattribute zu definieren welche einen vorgegeben Wertebereich haben. In diesem Fall verweist das MADefinitionLeaf-Objekt auf ein EDStructure-Objekt.
- Internal: Enumerationen welche nur internen Zwecken dienen sind mit Internal gekennzeichnet. Konkret wird dies eingesetzt um eine Enumeration zu verwalten, welche alle möglichen MAValue Typen verwaltet. Dies sind nebst MAValueString, MAValueInt, MAValueBool und MAValueImage auch alle Metadata-Enumerationen.

B.4.3 Umgesetztes Datenmodell

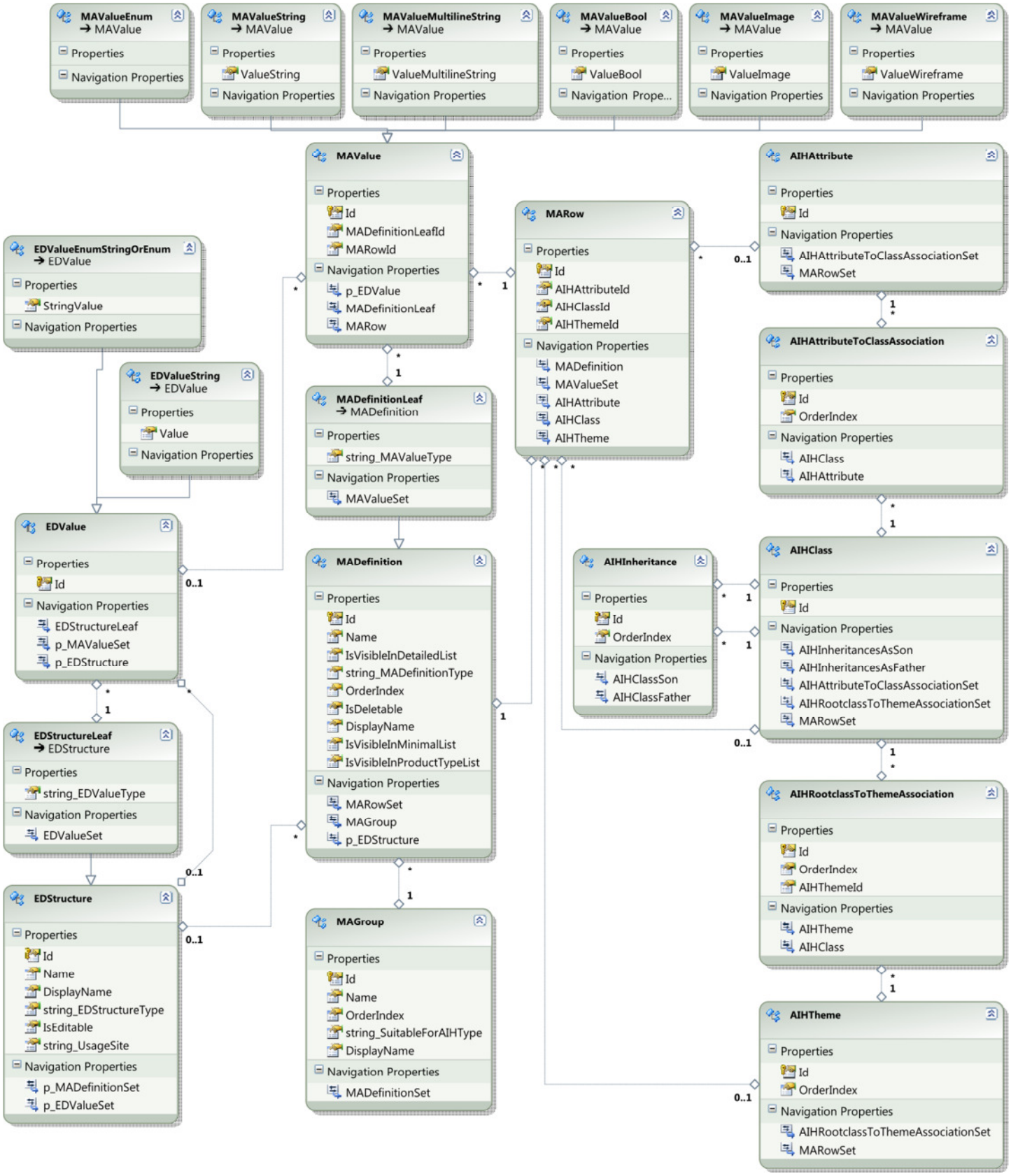


Abbildung 23: Umgesetztes Datenmodell

Unterschiede

Das umgesetzte Datenmodell weist einige Unterschiede zum konzeptionellen Datenmodell auf. Nachfolgend sind die Wichtigsten kurz erläutert.

Assoziationen auf Elternklasse

Wir haben an verschiedenen Stellen Vererbung in unserem Datenmodell. Damit sich dies nicht auf die Performance auswirkt, setzen wir die Strategie Table per Hierarchie ein. Der eingesetzte Datenbankgenerator ermöglicht allerdings keine Assoziationen auf Kindklassen. Davon betroffen sind folgende Assoziationen:

- EDStructure zu MADefinitionLeaf
- EDValue zu MAValueEnum
- EDStructure zu EDValueEnumStringOrEnum

Diese haben wir jeweils mit dem Präfix „p_“ bezeichnet. Im Code ist dieses Präfix nicht nötig: Wir haben in den partial class der jeweiligen Kind Entitäten den passenden Zugriff ermöglicht.

Komplexe Metaattribute

Leider konnten wir die komplexen Metaattribute nicht fertig stellen. Somit fehlt die Klasse MADefinitionComposite. Um diese Ergänzung ohne Refactoring der Datenbank zu ermöglichen haben wir die Klasse MARow bereits implementiert; diese würde aktuell noch nicht gebraucht werden.

Komplexe Enumerationen

Analog zu den komplexen Metaattributen fehlt auch die Klasse EDStructureComposite. Bei den Enumerationen fehlt auch die Klasse EDRow: Wir sehen keinen Use Case für Enumerationen welche aus ganzen Tabellen bestehen. Alle anderen Fälle lassen sich auch ohne EDRow umsetzen.

Andere Attribute

Diejenigen Attribute welche mit „string_“ beginnen zeigen an, dass die Entity Klasse im Programm jeweils ein Attribut besitzt, welches nicht eins zu eins auf die Datenbank abgebildet werden kann. Diese Attribute speichern zum einen Enum Typen und zum anderen Class Typen.

Mehr Attribute

Im konzeptionelle Datenmodell haben wir der Einfachheit halber einige Attribute nicht oder vereinfacht eingezeichnet. Diese sind für das Verständnis des Datenmodells nicht zentral. Um dem Leser das Unterschiede Suchen zu ersparen seien diese hier aufgelistet:

- Diverse OrderIndex Felder um die Reihenfolge zu verwalten. Speziell zu erwähnen sind die neuen Klassen AIHRootClassToThemeAssociation und AIHAttributeToClassAssociation welche nur zu diesem Zweck eingefügt wurden.
- Einige Klassen haben einen DisplayName erhalten; das Feld Name wird teils für interne Zwecke verwendet und muss somit unveränderbar sein.
- Für die MADefinitions braucht es Angaben, ob sie in den Listen Ansichten angezeigt werden sollen.
- IsDeletable bei MADefinition und IsEditable bei EDStructure geben an, wie weit Entities vom User manipuliert werden dürfen.

B.4.4 Design Prinzipien

B.4.4.1 ViewModelLocator

Verwaltet ViewModel

Die Tab Struktur wird vom MainWindow.xaml aufgebaut. Um nicht die ganze Struktur in ViewModels nachbilden zu müssen und die jeweils passenden ViewModel Objekte zwischen den Views zu übergeben, haben wir eine statische, zentrale Anlaufstelle: Den ViewModelLocator. Von ihm kann sich jede View die benötigten ViewModel direkt geben lassen. Der ViewModelLocator erstellt die ViewModel und sorgt dafür, dass alle mit denselben DataAccess und Messenger Instanzen arbeiten.

B.4.4.2 DataAccess

Zugriff auf Daten

Der DataAccess ist die zentrale Klasse des DAL. Über ihn lassen sich Daten aus der Datenbank lesen und in die Datenbank schreiben.

Events

Bei jeder Änderung der Daten löst der DataAccess einen entsprechenden Event aus. Dies ermöglicht den ViewModels bei Bedarf die aktuellen Daten zu laden. Es gibt folgende drei Events:

- ObjectCreated
- ObjectDeleted
- AttributeUpdated

Bei Änderungen welche mehrere Entitäten betreffen wird immer nur ein Event ausgelöst.

Änderungen sofort speichern

Die ViewModels sind dafür verantwortlich, sämtliche Änderungen welche der User vornimmt sofort in der Datenbank zu speichern. Somit werden andere interessierte ViewModel durch Events über die Änderungen informiert.

Meist binden die Views direkt auf Domain Objekte welche vom Entity Framework verwaltet werden. Die ViewModels müssen auch Änderungen an diesen Objekten überwachen und explizit über den DataAccess persistieren.

B.4.4.3 Messenger

MVVM Light

Als Message Bus setzen wir die Messenger Klasse aus dem MVVM Light Toolkit ein. Nur Objekte der obersten zwei Layer (ViewModels und Views) kommunizieren über den Message Bus.

Lose Kopplung

Die verschiedenen ViewModels kennen sich gegenseitig nicht. Änderungen auf einer Ansicht welche für andere Ansichten relevant sein könnten müssen per Message bekannt gegeben werden.

Ausnahmen bilden ViewModels welche zusammen für eine Ansicht verantwortlich sind. Für hierarchische Daten und Listen von Daten kann es nötig sein, dass ein ViewModel eine Liste von ViewModels verwaltet.

B.5 Schlussfolgerung

B.5.1 Ergebnisse

Relevante Daten identifiziert

Wir haben ein sinnvolles Set an Daten identifiziert, welche in der ersten Phase einer Produktstrukturierung erfasst werden müssen:

- Attribute welche die Produkte charakterisieren
- Klassen um Produkte zu abstrahieren und mit Attributen zu beschreiben
- Vererbungsbeziehungen zwischen den Klassen um die Produkte zu strukturieren
- Metaattribute zu Attributen und Klassen, diese müssen für die verschiedenen Produktstrukturierungsprojekte angepasst werden
- Enumerationen mit ihren Werten (z.B. ein Farbraum mit allen zugehörigen Farbedefinitionen)

Daten modellieren

Mit PSA haben wir das erste Programm entwickelt, welches die Produktstrukturierung in der ersten Phase unterstützt und strukturiert. In PSA können die relevanten Daten modelliert, visualisiert und explorativ erkundet werden. Die gesamte Produktstruktur kann abschliessend als XML exportiert werden und so in ein geeignetes Format transformiert werden.

Daten visualisieren

Um die Komplexität und den Umfang einer Produktstruktur zu bewältigen, bietet PSA verschiedene Visualisierungen an.

- Die Attribute, Klassen und Themen lassen sich inklusive ihrer Metadaten auflisten, sortieren und filtern.
- Die komplette Vererbungshierarchie wird in einer übersichtlichen Baumansicht einfach visualisiert. Die Klassen können in Themen strukturiert werden.
- Alternativ lässt sich ein UML Klassendiagramm generieren. Um die Übersicht zu behalten kann man auch nur einen Teil der Vererbungshierarchie mit einbeziehen.
- Eine einfache Auflistung aller Enumerationen und ihrer zugehörigen Werte.
- Für die einzelnen Produkte lassen sich Eingabemasken erzeugen, welche einen Eindruck über den Umfang der zu erfassenden Daten und den damit einhergehenden Aufwand bieten.

Daten erkunden

In den Listen Ansichten können die Daten nach verschiedenen Metaattributen oder Eigenschaften der Modellierung sortiert und gefiltert werden. So kann man z.B. schnell alle nicht verwendeten Attribute ausfindig machen.

Durch Markierungen in der Baum Ansicht können allfällige Schwächen der Modellierung aufgedeckt werden. Z.B. können alle Klassen welche zu viele gemeinsame Attribute haben markiert werden.

Einordnung

PSA ist das erste Programm welches die erste Phase der Produktstrukturierung abbildet. Es ersetzt bisherige Dokumentationsversuche in Excel oder anderen Adhoc Formaten.
Wie weit in einem Projekt die Modellierungsmöglichkeiten genügen, wird sich erst in einem Praxiseinsatz zeigen. Auf jeden Fall bietet PSA einen wertvollen

Beitrag zur Diskussion um die Strukturierung der Produktstrukturierung.

B.5.2 Ausblick

Diskussion

Heutige PCM bieten diverse Möglichkeiten Produkte über verschiedene Kanäle dem Kunden zu präsentieren. Konfiguriert werden sie allerdings in XML Files. Hier wäre es wünschenswert eine Anbindung einer Software wie PSA zu haben.

Je nach Wahl des konkreten PCM kann man auf dessen Stärken setzen und muss mit dessen Schwächen leben; diese Wahl muss wohl überlegt erfolgen. Hierfür ist es sinnvoll, die ersten Schritte der Produktstrukturierung unabhängig vom PCM durchzuführen. Es ist also eine Standalone Software nötig. Welche Merkmale der Produktstruktur allerdings als „Trockenübung“ ohne PCM Anbindung möglich und sinnvoll zu modellieren sind, muss sich noch zeigen.

Weitere Features

Es gibt noch einige Ideen für Features, welche in PSA integriert werden könnten. Hier der Anfang einer wahrscheinlich sehr langen Liste:

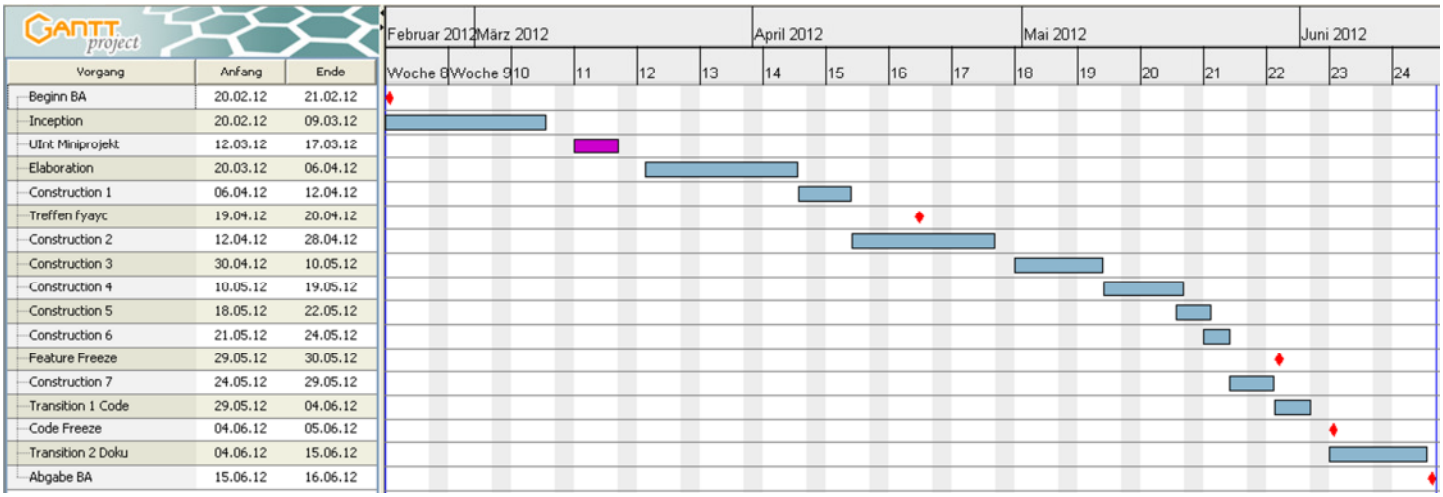
- Statistische Auswertung von Metadaten. So liesse sich z.B. ein Kostenschätzungsmodell implementieren.
- Komplexe Metaattribute und Enumerationen um Daten in ihrem Zusammenhang erfassen zu können.
- Tiefergehende Modellierung von korrelierenden Attributen: Z.B. Attribute mit mehreren Werten (Höhe x Breite), verschiedene Farbräume bei welchen Einträge miteinander zusammenhängen (Marketing Farben, Farben für Zubehör), …
- Navigationshierarchien Modellierung: Es bietet sich an, für verschiedene Kanäle angepasste Navigationshierarchien zu erstellen. Diese werden meist logisch verwandt sein mit der Attribut Vererbungshierarchie.
- …

Patterns

Es gibt Daten und Muster welche sich in vielen Projekten ähnlich sind oder gar wiederholen: Eine Artikel Klasse mit typischen Attributen wie z.B. Artikelnummer und Preis, Enumerationen von Farben, das Thema Lagerhaltung, etc. Solche Standard Patterns könnten auf Knopfdruck zur Verfügung gestellt werden.

C. Projektmanagement

C.1 Übersicht über den Projektverlauf



Vorgang	Anfang	Ende
Beginn BA	20.02.12	21.02.12
Inception	20.02.12	09.03.12
UInt Miniprojekt	12.03.12	17.03.12
Elaboration	20.03.12	06.04.12
Construction 1	06.04.12	12.04.12
Treffen fyayc	19.04.12	20.04.12
Construction 2	12.04.12	28.04.12
Construction 3	30.04.12	10.05.12
Construction 4	10.05.12	19.05.12
Construction 5	18.05.12	22.05.12
Construction 6	21.05.12	24.05.12
Feature Freeze	29.05.12	30.05.12
Construction 7	24.05.12	29.05.12
Transition 1 Code	29.05.12	04.06.12
Code Freeze	04.06.12	05.06.12
Transition 2 Doku	04.06.12	15.06.12
Abgabe BA	15.06.12	16.06.12

Abbildung 24: Ganttdiagramm Projektverlauf

Inception

Beim ersten Meeting versuchten wir das Ziel der Arbeit zu konkretisieren. Herr Keller schilderte uns das Problem, dass grosse UML Diagramme am Beamer kaum darzustellen sind. Weiter wünschte er, dass man Performanceprobleme möglichst schon in einem UML erkennen können sollte.

Ziel des Projekts war also, verschiedene Darstellungsprobleme anzugehen und Verbesserungen zu suchen.

In der Sitzung vom 08.03.2012 gab uns Herr Keller neue Ziele. Herr Keller war mit Herrn Möller bei einem neuen Kunden und schilderte uns aktuelle Probleme, für welche sie gerne eine bessere Lösung hätten. Von nun an ging es darum, eine Software für die Produktstrukturierung zu entwickeln.

UInt Miniprojekt

Mit Soll-Scenarios und Wireframes haben wir eine erste Anforderungsanalyse durchgeführt und einen ersten, einfachen Papierprototypen von PSA erstellt. Die komplette Dokumentation des Miniprojekts ist im Anhang zu finden.

Elaboration

Von nun an verlief das Projekt etwas ruhiger. Nachfolgend werden jeweils die wichtigsten Arbeitspakete und Themen aufgelistet:

- Recherche Technologien
- Beginn Umsetzung des Prototyps aus UInt Miniprojekt
- Software Architektur
- Datenmodell

Construction 1

- Attribut Liste
- Metaattribute
- UML-Editor

Construction 2

- Neue Navigation
- Fertigstellen des Programms mit Wireframes
- Treffen bei foryouandyourcustomers

Construction 3

- Treffen mit Gegenleser Herrn Rudin
- Refactorings, Arbeiten beenden, Ansicht nicht neu aufbauen
- Alternative Darstellung Vererbung überlegen

Construction 4

- Alternative Darstellung Vererbung umsetzen (Tree mit Spalten)
- Inkl. Drag & Drop
- Erste Filter

Construction 5

- Vererbungshierarchie Tab beenden
- Inkl. Mehrfachvererbung, Reihenfolge der Klassen und Attribute, Drag & Drop, Icons

Construction 6

- KonfigurationsTab (Logo setzen, Backup)
- Enumerationen modellieren

Construction 7

- Performance
- (weitere) Filter
- Metadatenstrukturierung
- Eingabemasken
- Export
- Import

Transition 1

- Code abgabebereit machen
- Testen
- Debuggen

Transition 2

- Dokumentation beenden

C.1.1 Vergleich mit Planung

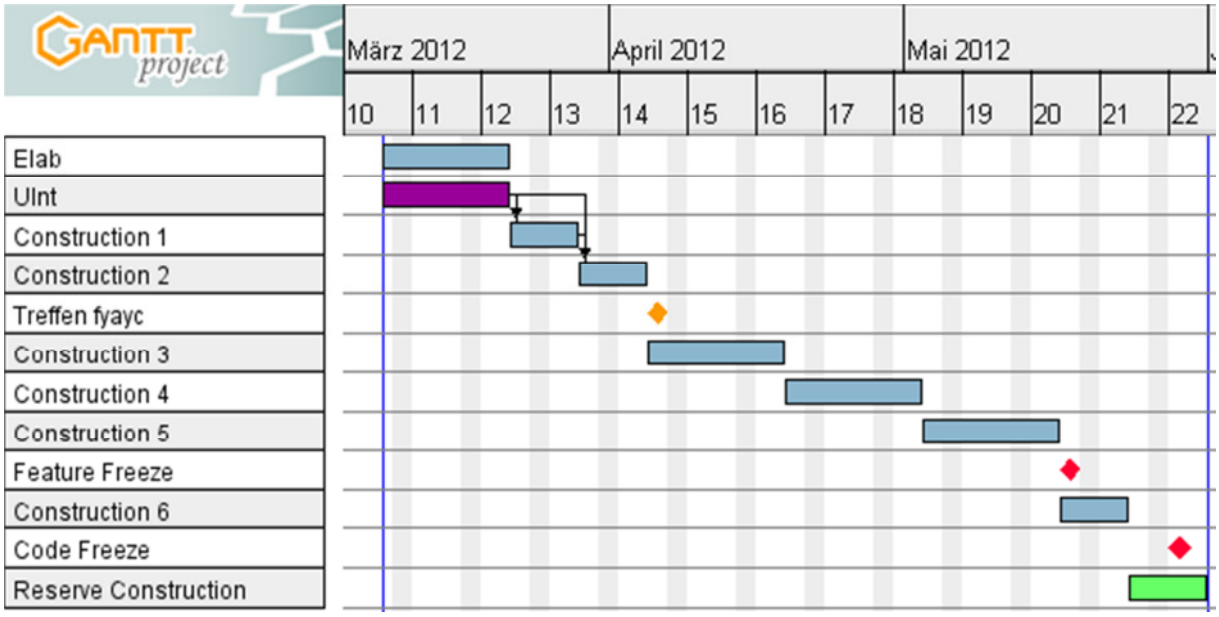


Abbildung 25: Ganttdiagram Planung Construction

UInt Miniprojekt

Da Herr Keller in der Kalenderwoche 12 in den Ferien war, führten wir das Userinterface Miniprojekt komplett in einer Woche durch. Dafür haben wir

eine Woche später mit der Elaboration begonnen.

Treffen fyayc

Das Treffen mit Jonathan Möller haben wir um zwei Wochen Verschoben. Dafür konnten wir einen Prototyp vorstellen, bei welchem die ganze Navigationsstruktur fertig war. Die nicht implementierten Features wurden mit Wireframes vorgestellt. Damit hatten wir eine sehr gute Diskussionsgrundlage.

Feature Freeze, Code Freeze

In der Construction 4 (Kalenderwoche 19) haben wir entschieden, die Reservewoche welche für die Construction eingeplant war, einzusetzen. Dadurch haben sich die Meilensteine Feature Freeze und Code Freeze um eine Woche verschoben.

C.2 Zeitauswertung

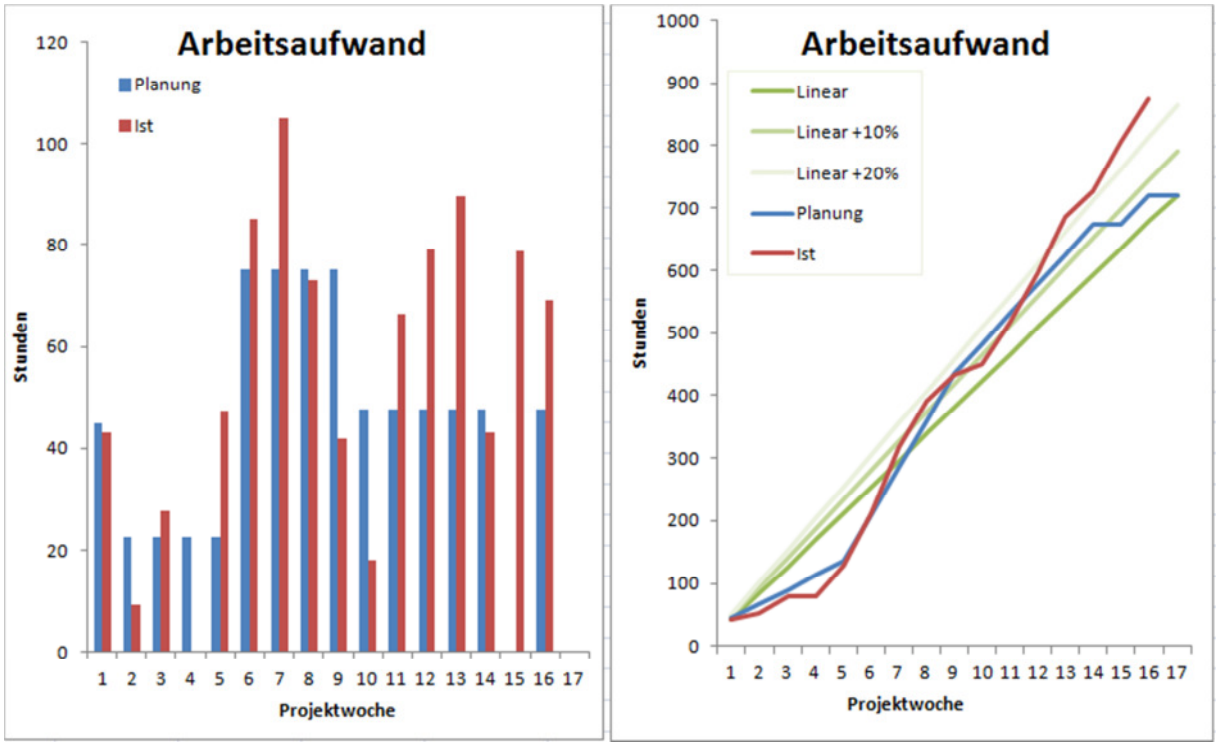


Abbildung 26: Diagramm Arbeitsaufwand

Wochen 2 – 5

In den Projektwochen 2 bis 5 hatten wir wenig Zeitaufwand geplant. Zum einen nahmen wir an einem Wettbewerb von Microsoft teil (You Make IT Smart), zum anderen bearbeiteten wir das Userinterface 2 Miniprojekt.

Zeitbudget

Für die Bachelorarbeit werden 12 ECTS Punkte an das Studium angerechnet. Laut Bologna Regeln sind somit pro Person mindestens 360 Stunden zu investieren. Da wir beide motiviert und ehrgeizig sind, planten wir mit einem Budget von bis zu 864 Stunden, dies entspricht 20% mehr Aufwand. In den letzten Iterationen der Construction Phase verloren wir das Budget ein bisschen aus den Augen: Die vielen kleinen Ideen und Verbesserungen die wir noch sahen, wollten wir auch noch umsetzen. Bis jetzt (Ende Projektwoche 16) haben wir 875 Stunden gearbeitet. Wir werden somit unser maximales Budget um etwa 5 Prozentpunkte sprengen.

C.3 Organisation

Iterativ

Unsere Aufgabenstellung war sehr offen. Es gab keine Feature Liste. Um dieses Projekt zu bearbeiten drängten sich agile Entwicklungsmethoden geradezu auf. Wir liessen uns insbesondere von RUP und Scrum inspirieren, gingen aber nicht streng nach einem dieser Modelle vor. Das wichtigste Charakteristikum war das iterative Vorgehen.

4-Augen-Prinzip

Wir haben verschiedene Verantwortungen auf Rollen aufgeteilt. Die Idee war nicht eine ausschliessliche Verantwortung. Es ging eher darum, dass die jeweilige Person ein besonderes Augenmerk auf ihre Themen werfen sollte. Entscheidungen sollten immer von beiden gemeinsam gefällt werden und – nach dem 4-Augen-Prinzip – sollten auch immer beide einen Überblick über die ganze Arbeit haben.

Rollen

- Christoph Rosenberger:
  - Projekt Manager
  - Risiko Manager
  - Qualitäts Manager
- Michael Steiner
  - Project Owner
  - Problem Manager

Diese Rollenaufteilung hat eigentlich ganz gut funktioniert. Insbesondere zu Projektbeginn hat sie uns geholfen, uns zu Recht zu finden. Im Projektverlauf wurde sie immer weniger wichtig.

C.4 Qualität

Qualitäts-standard

Unser Auftrag war es einen Prototypen zu entwickeln, welcher die Konzepte und Möglichkeiten aufzeigt. Natürlich sollte er auch bedienbar sein, so dass man ihn einsetzen kann. Deshalb haben wir von Beginn an fortlaufend sauber programmiert. Lesbarer Code ist uns sehr wichtig. Wir führten auch immer wieder kleine Refactorings durch, so dass immer Ordnung herrschte. Dies hat sich ausbezahlt. Wir mussten am Schluss nicht mehr viel Zeit fürs Debugging aufwenden.

Test-protokoll

Unser Code wurde primär durch ausführliche Benutzertests getestet, welche am Projektende durchgeführt wurden. In einem 14seitigen Testprotokoll mit 153 Tests wird dem Tester genau vorgeschrieben, welche Aktionen er testen muss und welche Resultate zu erwarten sind. Dadurch können wir die Qualität über alle Layer hinweg testen. Der erste Durchlauf wies uns noch auf einige kleine Probleme hin. Im zweiten Durchlauf hatten wir bereits keine Fehler mehr.

Unittests

Mit den Unittests decken wir nur die wichtigsten Funktionen im DAL ab.

Hierarchy	Not Covered (Blocks)	Not Covered (% Blocks)	Covered (Blocks)	Covered (% Blocks)
Michael@NBMICHAEL 2012-06-15 08:15:46	2946	61.61%	1836	38.39%
PSA._40_DAL.dll	2946	61.61%	1836	38.39%
{ } PSA._40_DAL	2138	58.93%	1490	41.07%
{ } PSA._40_DAL.MAMetaAttributableFilter	808	70.02%	346	29.98%

Abbildung 27 Code Coverage

Codezeilen

Durch unseren sehr generischen Code kommt unser Programm mit relativ wenig Codezeilen aus:

Views (exkl. Xaml)	568
ViewModels	1953
BusinessLogic	616
DAL	2250
	5387

C.5 Technologien / Tools

Während der Inception haben wir für die Zeiterfassung mite verwendet.

http://mite.yo.lk/

Ab der Elaboration haben wir Redmine eingesetzt für die Planung der Iterationen mit Arbeitspaketen, als Backlog und für die Zeitaufschreibung.

http://www.redmine.org/

Um Gant Diagramme unabhängig von Redmine zu erstellen kam GanttProject zum Einsatz.

http://www.ganttproject.biz/

Während Heimarbeitssessions haben wir unsere Kommunikation mit dem TeamViewer unterstützt.

http://www.teamviewer.com/de/

Für den unkomplizierten Austausch von Links, um gemeinsam Texte zu schreiben und einiges mehr haben wir Google Docs verwendet.

http://docs.google.com

Für die Versionsverwaltung haben wir Subversion eingesetzt.

http://subversion.apache.org/

Um den Umgang mit Subversion zu erleichtern kam TortoiseSVN zum Einsatz.

http://tortoisesvn.net/

C.6 Reflexion: Lernen aus Problemen

Grösstes Problem

Wir haben unsere Bachelorarbeit in einem Themengebiet gemacht, in welchem wir uns nur oberflächlich auskennen. Wer einmal mit Jonathan Möller – Gründer von foryouandyourcustomers – ein faszinierendes Gespräch hatte, weiss wie breit und vielschichtig die Themen Multichannel und Produktstrukturierung sind. Dass wir innerhalb dieser Arbeit selbst zu Experten auf diesem Gebiet werden konnten, war wohl kaum möglich. Daher waren wir auf möglichst viel Input angewiesen. Aus rechtlichen Gründen war es leider nicht möglich, dass wir zu Beginn des Projekts Beispieldaten vergangener Projekte hätten haben können.

Grösstes logisches Problem

An der Erstellung des Datenmodells haben wir uns beinahe die Zähne ausgebissen. In einem ersten Schritt hatten wir knapp verstanden, wie wir die Metaattribut Definitionen und ihre Werte verwalten konnten; auch diejenigen mit mehreren Spalten und Zeilen. Dann sahen wir, dass dieselbe Struktur auch für die Enumerationen sinnvoll war. Und dann gab es da auch noch die Attribute der Metaattribute, also der Metametaattribute, welche auch in diese Struktur gepasst hätten… Auf der Suche nach einer schönen, verständlichen, generischen, pragmatischen Lösung drehten sich unsere Gedanken schnell in einer Rekursion ohne Abbruchbedingung. Zum Glück wird der Mensch irgendwann müde, sonst würden unsere Gedanken heute noch Karussell fahren.

Grösstes technisches Problem

Wir hatten zwar einige Mühe mit WPF; insbesondere eine TreeView mit Spalten welche untereinander angeordnet sein sollten bescherte uns einige Stunden mit Googlen. Das noch grössere, dafür extrem lehrreiche, technische Problem lag bei der Performance.

Als wir das erste Mal eine grössere Menge an Daten geladen hatten, ging es 20 Minuten um PSA zu starten. Mit dem in Visual Studio integrierten Profiler hatten wir ohne viel Arbeitsaufwand die schuldige Funktion gefunden. Unglücklicherweise mussten wir ein paar Zeilen umprogrammieren, so dass nicht eine Funktion des .Net Frameworks als Zeitintensivste angezeigt wurde: nach 12 Versuchen sind 4 Stunden ohne viel Arbeit vorbei… Den allergrössten Teil der Zeit wurde von einer Funktion verbraucht: Sie muss beim Start für alle AIHAttribute viele MAValue laden.

Zuerst versuchten wir alle Daten aufs Mal zu laden: Die Logik ein bisschen anpassen und Eager Loading aktivieren - brachte aber keinen Erfolg.

Als nächstes wollten wir das von LINQ generierte SQL analysieren. Dabei lernten wir den Unterschied von IQueriable und IEnumerable kennen: Das erste wird zu SQL übersetzt, das zweite lädt die Daten und arbeitet im Memory die Listen ab. Aber auch der nicht ganz triviale Wechsel auf IQueryable brachte noch nicht die gewünschte Performance.

Die nun analysierbare SQL Query wies Verbesserungspotenzial auf. Durch Umstellen der LINQ Befehle erzielten wir die nächste Verbesserung.

Nun sah das generierte SQL eigentlich ganz vernünftig aus: Ausser dass alle unsere Kindklassen des MAValue gejoint werden mussten. Wir waren davon

ausgegangen, dass die einzelnen Tabellen nicht sehr viele Records haben und somit diese Joins nicht so schlimm sein konnten. Was sind schon ein paar hundert Zeilen gegenüber einer ausgewachsenen Business Applikation? Da wir aber keinen anderen Schuldigen ausmachen konnten, versuchten wir diese Joins zu umgehen.

Wir konfigurierten Entity Framework mit dem Model-First Ansatz: D.h. wir nutzten den UML ähnlichen Diagramm Editor und liessen sowohl die Datenbank als auch die C# Klassen generieren. Leider unterstützte der in Visual Studio integrierte Generator nur eine Mapping Strategie für Vererbung: Table per Type. Um die Joins zu eliminieren brauchten wir aber ein Table per Hierarchie Mapping. Dies in eigenen T4 Templates umzusetzen lag allerdings (leider) bei weitem nicht im Zeitplan. Glücklicherweise konnten wir im Internet ein Tool finden, welches ein entsprechendes Template mitbrachte (auch wenn dies einige Unschönheiten mit sich brachte: So können Kindklassen keine eigenen Assoziationen mehr definieren).

Nun endlich hatten wir die gewünschte Performance und müssen einsehen: Herr Keller ist besser als Microsoft. Entity Framework unterstützt als einzige Mapping Strategie Table per Type. Herr Keller unterstützt als einzige Mapping Strategie Table per Hierarchie.

Grösster Fehler

Zu Beginn des Projekts analysierten wir die Anforderungen an die Metaattribute. Wir stellten fest, dass für manche Fälle nicht nur einzelne Werte erfasst werden müssen. So braucht es z.B. für eine Rechteverwaltung eine ganze Tabelle: Für verschiedene Rollen müssen Schreib- und Leserecht verwaltet werden.

Die Komplexität des Datenmodells wäre gross genug gewesen, wenn wir nur einfache Werte unterstützt hätten. Durch diese komplexen Metaattribute wurde die Aufgabe entsprechend schwerer.

Die einfachen Metaattribute sind für die meisten Probleme ausreichend. Es ist fraglich, ob sich durch die anderen Fälle der grosse Aufwand rechtfertigen liesse.

Leider konnten wir die komplexen Metaattribute nicht in der geplanten Iteration, es war eine der ersten, fertig stellen. In den folgenden Iterationen war diese Funktionalität meistens in der Planung. Da aber der zusätzliche Nutzen nicht gross genug gewesen wäre, war die Priorität dieses Features nie hoch genug, als dass wir es fertig gestellt hätten.

So haben wir zwar den schwierigsten Teil der Arbeit erledigt, der Kunde hat aber nichts davon. KISS: Keep It Simple And Stupid. Brachte man uns schon im ersten Jahr bei. Bei Bedarf hätten wir immer noch eine schwierigere Variante umsetzen können.

C.7 Reflexion: Lernen aus Erfolgen

Teamarbeit

Dass wir grundsätzlich als Team funktionieren, wussten wir aus vergangen Projekten. Wichtig war uns aber, dass wir unsere Teamarbeit optimieren konnten. Wir kennen unsere gegenseitigen Stärken und Schwächen und konnten einander so gut unterstützen, voneinander lernen und profitieren.

Arbeitsmethodik

Wir experimentierten mit verschiedenen Arbeitsmethoden. Zum einen arbeiteten wir viel per Teamviewer. Dies sparte uns den Arbeitsweg und ausserdem konnten wir so die gewohnte Infrastruktur zu Hause nutzen. Per Telefon konnten wir die meisten Probleme lösen und auch gute Diskussionen führen. Auch unterstützte uns die Zeichenfunktion des Teamviewers.

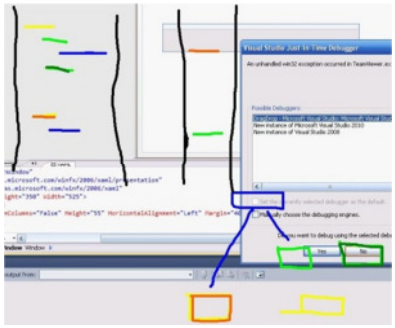


Abbildung 28: Bildschirmzeichnung 1

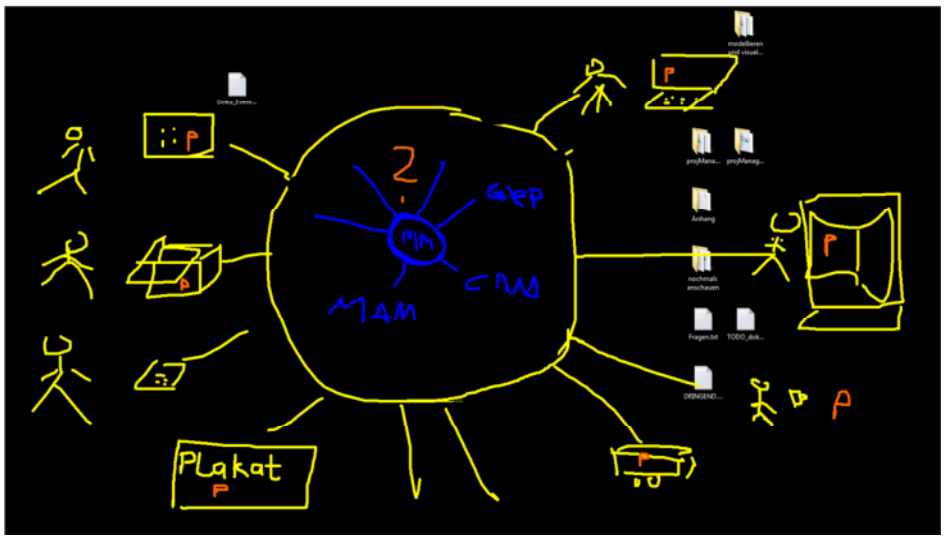


Abbildung 29: Bildschirmzeichnung 2

Wireframes

Da wir zu Beginn nicht wussten, wie unser Produkt aussehen und was es können sollte, beschäftigten wir uns mit dieser Frage zusätzlich in unserem UInt 2 Miniprojekt. Dort mussten wir Wireframes, welche die Funktionalität unseres zukünftigen Programms beschreiben, entwerfen. Für uns war das eine überraschend positive Erfahrung. Wir, aber auch unser Kunde, mussten sehr konkret werden. So machten wir einen grossen Schritt nach vorne. Da wir so begeistert waren von dieser Idee beschlossen wir, für ein besseres Feeling die Wireframes in ein lauffähiges Programm einzubinden. So implementierten wir anschliessend Wireframe um Wireframe, bis ein voll funktionsfähiges Programm entstand. Geplante Funktionalität, welche aber zurzeit nicht benötigt wird, haben wir als Wireframes im Programm stehen lassen, damit man gleich sieht was noch alles möglich wäre.

Auch wurde unser Meeting mit dem Kunden durch die Wireframes positiv unterstützt. Wir mussten mit ihm nicht darüber diskutieren, was wir noch alles machen werden, sondern er konnte auf einen Blick sehen was wir alles geplant haben und wie dies funktionieren könnte.

Architektur

Unsere Architektur kostete uns zu Beginn viel Zeit und wir hatten dementsprechend etwas länger bis wir ein Tool „zum Anfassen“ hatten. Im späteren Projektverlauf kam uns dies aber mehr als einmal zu gute. Unsere Architektur überlebte alle Veränderungen und konnte immer einfach erweitert werden.

D.Verzeichnisse

D.1 Glossar

ECTS	European Credit Transfer System: Europäisches Punktesystem welches mit der Bologna Reform eingeführt wurde. Für einen Bachelorabschluss an der HSR braucht es 180 Punkte. Die Bachelorarbeit wird mit 12 Punkten angerechnet. Ein Punkt entspricht einer Arbeitsleistung von 30 Stunden.
UInt 2	Userinterfaces 2 ist ein Aufbau Informatik Modul an der HSR. Teil dieses Moduls ist ein Miniprojekt in welchem mittels User Centered Design ein UI entworfen und mit einem Papierprototyp getestet werden muss.
PIM	Product Information Management
PCM	Product Content Management: System in welchem Produktstruktur abgelegt ist. (Die wichtigsten: Heiler, Hybris, Stibo, Riversand)
WPF	Windows Presentation Framework
MVVM	Das „Model View ViewModel“ Pattern dient der sauberen Implementierung des GUI.
EF	Entity Framework ist ein OR-Mapper von Microsoft.
OR-Mapper	Object Relational Mapper. Übersetzt von der Objektorientierten Welt in die Welt der relationalen Datenbanken.

D.2 Abbildungsverzeichnis

D.3 Quellenverzeichnis

D.3.1 Bilder

Für unsere Beispieldaten haben wir Bilder aus dem Webshop von opo.ch verwendet.

D.3.2 Code

- Delete an image bound to a control
  http://stackoverflow.com/questions/690150/delete-an-image-bound-to-a-control
- MarginSetter
  http://blogs.microsoft.co.il/blogs/eladkatz/archive/2011/05/29/what-is-the-easiest-way-to-set-spacing-between-items-in-stackpanel.aspx
- Gridsplitter
  http://wpf.2000things.com/2011/12/30/462-drawing-a-better-looking-gridsplitter/
- DataGridView
  http://blogs.msdn.com/b/markrideout/archive/2006/01/08/510700.aspx
- Drag & Drop
  http://www.wpftutorial.net/draganddrop.html

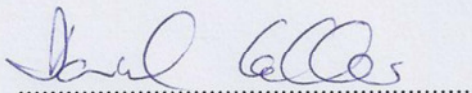
E. Anhang

Anhang E1
Aufgabenstellung

Aufgabenstellung

Visualisierung von komplexen Produktstrukturen

Studiengang	Informatik (I)
Semester	FS 2012 (20.02.2012-16.09.2012)
Durchführung	Bachelorarbeit
Fachrichtung	Software
Institut	IFS: Institut für Software
Gruppe	Christoph Rosenberger Michael Steiner
Betreuer	Daniel Keller
Gegenleser	Hans Rudin
Experte	Rudolf Mattmann, Mettler-Toledo
Industriepartner	foryouandyourcustomers.com
Aufgabenstellung	Im Produkt-Informationssystem (PIM) einer grossen Firma (Bsp. Migros, Hornbach) stecken in erster Linie Daten über die Produkte und ihre Eigenschaften (Preis, Farbe, Gewicht, Lieferant...). Dazu kommen aber noch Klassifizierungen ('Socken', 'Tierfutter', mit Vererbung) und Kategorisierungen ('extra leicht', 'pastellfarbig', 'Sommerkollektion 2011', 'ist Zubehör zu X', 'ist upgrade zu Y', 'ist Ersatz für Z' ...). Das macht die Sache schon recht komplex, vor allem, wenn Zugriffs-Effizienz eine Rolle spielt. Es soll Software entwickelt werden, welche die Datenstrukturen mitsamt Klassifizierung und Kategorisierung visualisiert und dokumentiert.



.....

Daniel Keller

Anhang E2
Eigenständigkeitserklärung

Erklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben.
- das wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Rapperswil, 15.06.2012

Michael Steiner

Christoph Rosenberger

Anhang E3
Persönliche Berichte

Michael Steiner

Vorgeschichte

Dank unserer Semesterarbeit hatten wir einen Einblick in die Welt der Produktstrukturierung. Dadurch wussten wir immerhin schon einmal wo der Schuh drückte, jedoch nicht, wie wir das Problem in den Griff bekommen könnten.

Neue Herausforderung

Zu Beginn unserer Arbeit wusste niemand, wie unser Resultat aussehen würde. Wir wussten auch nicht, wie wir den Kunden am besten unterstützen können. Der Kunde selbst wusste es auch nicht so genau. So mussten wir uns gemeinsam durchschlagen und uns überlegen, wo wir ihm bei der Produktstrukturierung helfen können. Dies war für mich eine neue und unbekannte Herausforderung. Bis anhin wussten meine Kunden immer was sie wollten. Durch Reflektion konnte ich aus dieser Situation aber viel Lernen.

Wireframes

Erkenntnis Nummer zwei war, dass konkret werden eine gute Sache ist. Wir mussten uns überlegen, was dem Kunden von Nutzen sein könnte. Im Uint2 Miniprojekt entwickelten wir dann Wireframes, welche die Masken für die Datenerfassung (Attribute, Klassen und Metaattribute) zeigten. Diese Masken waren sehr einfach, doch auf die Idee zu kommen genau solche Masken zu erstellen eben nicht. Jetzt konnten wir mit dem Kunden konkret sprechen und ihm erklären was er jetzt mit dem Tool alles machen könnte und was nicht. So hatten beide Seiten ein klareres Bild und es entstanden wieder neue Ideen.

Mehrwert

Unsere Aufgabe war zuerst das Visualisieren von Produktstrukturen. Da der Kunde leider keine Daten hatte, mussten wir zuerst eine Möglichkeit schaffen, Daten zu erfassen. Als wir diesen Schritt machten, merkten wir, dass ein grosser Mehrwert für den Kunden in diesen Modellierungen steckt.

Erste Version

In der Mitte des Projektes konnten wir unserem Kunden dann auch eine erste Version präsentieren. Das Bild, was unsere Software können muss, wurde immer klarer. Dadurch bekamen wir auch immer mehr Freude an dem Projekt und so kam es dazu, dass wir uns entschlossen, unsere Reserve für weitere Programmierarbeiten einzusetzen.

Architektur

Es war am Anfang nicht leicht eine gute Architektur zu finden. Wir hatten viele Diskussionen darüber, wie man es am besten machen könnte. Das Problem war, dass wir nicht wussten, was noch alles auf uns zukommen wird. Man muss dazu vielleicht auch sagen, dass die Mitarbeiter von fyayc sehr schnell neue Ideen entwickeln können. Es freute mich sehr, dass sich unser Aufwand gelohnt hatte. Unsere Architektur „überlebte“ alle neuen Features und wir konnten so definitiv viel Zeit sparen. Dies hatte auch grosse Auswirkungen auf unser Voranschreiten: Hatten wir unseren Grundbaustein einmal gelegt, konnten wir extrem schnell und einfach neue Features implementieren. Dies führte, nach einem harzigen Start, zu vielen kleinen Erfolgserlebnissen und schlussendlich zu einem super Programm.

Christoph Rosenberger

Ich bin fertig.

Nun ist auch die Dokumentation fertig gestellt. Ich habe in den letzten Tagen so viel geschrieben, es ist schon fast alles gesagt. In ein paar Tagen, Wochen vielleicht auch Monaten würde meine Reflexion wohl anders aussehen. Heute bin ich erst einmal glücklich, dass wir den Abgabetermin für die Arbeit einhalten können. Und dies mit einem überzeugenden Resultat: Einer anständigen Dokumentation und einem coolen Programm.

Unnötiges Unwohlsein

Natürlich, man könnte immer etwas besser machen. Im Nachhinein sind die Fehler so einfach und offensichtlich. Natürlich, es könnte immer etwas besser laufen. Im Nachhinein sieht man aber, dass auch erschwerende Umstände die gute Lösung nicht zu verhindern vermögen.

Auch zu Beginn dieser Arbeit fühlte ich mich nicht wohl. Was machen wir überhaupt? Was ist das Ziel der Arbeit? Was wird von uns erwartet? Vielleicht kommt die Lockerheit und Zuversicht selbst in nebulösen Abschnitten eines Projekts mit der Erfahrung. Ich hoffe es. Ich habe ja auch in dieser Arbeit gesehen: Es kommt schon gut.

Natürlich ist es schwierig für ein Einsatzgebiet zu entwickeln, welches man nicht sehr gut kennt, in welchem man nicht Experte ist. Da ist man auf fundierten Input angewiesen, sogar davon abhängig. Natürlich hätten uns Beispieldaten zu Beginn des Projekts geholfen. Natürlich, es läuft nicht immer alles optimal, möglichst einfach, nach Wunsch. Trotzdem ist es möglich ein super Resultat zu erzielen. Wie diese Arbeit zeigt.

Optimistisch, konkret, einfach

Eine Taktik mit der Ungewissheit, der Unsicherheit umzugehen ist der Schritt ins Konkrete. Die Gedanken ausformulieren. Die Gedanken aufschreiben. Die Gedanken aufzeichnen. Und wenn die Diskussion wieder stockt: Eine erste einfache Version umsetzen, im laufenden Programm ausprobieren.

„Eine erste einfache Version“: besonders das „einfach“ sollte man sich aber zu Herzen nehmen. The Bigest Bang For The Bucks. Im ersten Schritt nur die einfachen Features einer Idee umsetzen, die schwierigen Teile mal weglassen. Sobald das Programm läuft sieht man viel besser, ob die schwierigen Teile überhaupt nötig sind. In der optimistischen Annahme, dass man man sich vor der Lösung der Probleme drücken kann, sollte man ruhig auch grössere Refactorings, allenfalls soger inklusive der Datenbank, in Kauf nehmen.

Danke Michael

Schlussendlich hat diese Arbeit ein super Resultat hervorgebracht, einen Mehrwert erzeugt. Aber die erfreulichste Erkenntnis ist der Beweis, wie gut ein Team arbeiten kann. Wie schön es ist, wenn man sich auf seinen Teampartner verlassen kann. Wie sehr man sich auf die Nerven gehen kann und sich trotzdem gut versteht. Wie schnell und qualitativ gut man arbeiten kann, wenn zwei Köpfe zusammen an einem Strick ziehen. Von daher: DANKE MICHAEL!

Danke HSR

Wenn ich schon beim Danken bin: Ein Dank auch an die HSR, und natürlich an Herrn Keller. Wenn ich sehe, wie ich vor dem Studium gearbeitet habe, meine

Projekte organisiert habe, wie mein Code ausgesehen hat, ich habe wirklich etwas gelernt. Und das in einer angenehmen Umgebung. Danke HSR.

Danke. An Mich.

Und der letzte Dank geht an mich. Dafür dass nun auch meine letzten Zeilen hier geschrieben sind. Nein. Eigentlich geht mein letzter Dank an meine Eltern für die super Unterstützung! Danke Mami! Danke Papi!

Anhang E4
Testprotokolle

Test Durchführung #1

Setup

Gestartet wird mit einer leeren Datenbank.

Durchführer	Michael Steiner
Datum	29.05.2012
System	Windows XP 32 Bit
.Net Version	4.0.30319

Testresultat Legende

	Test erfolgreich
	Test nicht erfolgreich, Status Unkritisch
	Test nicht erfolgreich, Status Kritisch

Testprotokoll

Daten Strukturierung

Attribute

Nr.	Aktion	Soll	Ist	Status
01.01	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.02	Name „Gewicht“ eingeben	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.03	Beschreibung „Gewicht der Kamera“ eingeben			Ok
01.04	Einheit „Kilogramm“ wählen	Einheit in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.05	Datentyp „Dezimalzahl“ wählen	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.06	Beispieldaten „0.57“			Ok
01.07	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.08	Name „Beschreibung“ eingeben	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.09	Einheit „keine“ wählen	Einheit in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.10	Unterscheidet Variation: Nie	Unterscheidet Variation in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.11	Datentyp „Mehrzeiliger Text“	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.12	Internationalisierbar: ja	Internationalisierbar in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.13	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.14	Name „Produktname“	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.15	Datentyp „Text mit Platzhalter“	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.16	Unterscheidet Variation: Immer	Unterscheidet Variation in der Tabelle wird		Ok

Christoph Rosenberger
Michael Steiner

		aktualisiert beim Verlassen des Feldes		
01.17	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.18	Name: Bildsensor	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.19	Datentyp: Ganzzahl	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.20	Metadatenstrukturierung → Enumerationen → Einheit → Megapixel erfassen	Megapixel kann der Enumeration hinzugefügt werden und wird angezeigt		Ok
01.21	Zurück zum Attribut Bildsensor und die Einheit „Megapixel“ hinzufügen	Megapixel steht jetzt in dem Dropdown und kann gewählt werden, Einheit in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.22	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.23	Name: Jahr	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.24	Datentyp: Datum	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.25	Beschreibung: Erscheinungsdatum			Ok
01.26	Auf Suchen Tab wechseln	Tab wechselt		Ok
01.27	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld	Metaattribut Tab wird nicht geöffnet (Usability)	Not Ok
01.28	Name: Shots	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.29	Beschreibung: Beispielbild aufgenommen mit diesem Produkt			Ok
01.30	Datentyp: Binärdaten	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.31	Auf Suchen Tab wechseln	Tab wechselt		Ok
01.32	Filter „Ähnliche Namen anwenden“	Attributliste wird leer		Ok
01.33	Auf Metaattribute Tab wechseln	Tab wechselt		Ok
01.34	Attribut erfassen	Filter wird entfernt, Neues, leeres Attribut		Ok

		Cursor im Namensfeld	
01.35	Name: Schwenkbares Display	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.36	Datentyp: Boolean	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.37	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld	Ok
01.38	Name: Grösse	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.39	Beschreibung: Gibt an ob Spiegelreflex oder Kompaktkamera		Ok

Themen (Daten Strukturierung → Themen)

Nr.	Aktion	Soll	Ist	Status
02.01	Thema erfassen	Neues, leeres Thema Cursor im Namensfeld		Ok
02.02	Name: Produkte	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok

Klassen (Daten Strukturierung → Klassen)

Nr.	Aktion	Soll	Ist	Status
03.01	Button Klasse erstellen muss „Enabled“ sein			Ok
03.02	Button Klasse löschen muss „Disabled sein“			Ok
03.03	Neue Klasse erstellen			Ok
03.04	Name: „Foto Kamera“ erfassen	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok

Vererbungshierarchie: Drag & Drop

Nr.	Aktion	Soll	Ist	Status
04.02	Auf Vererbungshierarchie Tab wechseln	Folgende Buttons müssen Disabled sein: • Beziehung entfernen • Knoten nach oben schieben • Knoten nach unten schieben Tab „Klassen“ muss geöffnet sein und die		Ok

		Foto Kamera selektiert Der Filter muss leer sein Die Metaattribute der Foto Kamera Klasse müssen geöffnet sein Das Thema „Produkt“ muss im Baum ersichtlich sein		
04.03	Die Klasse Foto Kamera per Drag & Drop auf PSA loslassen	Nichts passiert		Ok
04.04	Die Klasse Foto Kamera per Drag & Drop auf die weisse Fläche in der Baumstruktur ziehen	Nichts passiert		Ok
04.05	Die Klasse Foto Kamera per Drag & Drop auf den Filter ziehen und loslassen	Nichts passiert		Ok
04.06	Die Klasse Foto Kamera per Drag & Drop auf das Thema Produkte ziehen und loslassen	Drop Symbol erscheint und Klasse wird unterhalb des Themas angezeigt		Ok
04.07	In den Attribut Tab wechseln			Ok
04.08	Jedes Attribut auf die Klasse ziehen (Alphabetisch sortiert beginnen, alle ausser Shots)	Nichts passiert		Ok
04.09	Jedes Attribut auf die Attributspalte der Klasse „Foto Kamera“ ziehen (Alphabetisch sortiert beginnen, alle ausser Shots)	Attribute werden der Klasse hinzugefügt		Ok
04.10	Attribut Shots auf das Attribut Beschreibung im Baum ziehen	Shots wird hinter der Beschreibung eingefügt		Ok
04.11	Zum Klassen Tab wechseln			Ok
04.12	Zwei Klassen erfassen: Spiegelreflex, Kompakt und Canon EOS 5D MKIII			Ok
04.13	Die Klasse Spiegelreflex auf die Klasse „Foto Kamera“ im Baum ziehen	Die Klasse wird unten angefügt		Ok
04.14	Die Klasse Kompakt auf die Klasse „Spiegelreflex“ im Baum ziehen	Die Klasse wird unten angefügt		Ok
04.15	Die Klasse Kompakt im Baum auf die Klasse Foto Kamera ziehen	Kompakt ist jetzt auf gleicher Höhe (vererbungstiefe) wie Spiegelreflex Kompakt ist markiert		Ok
04.16	Button „Klasse nach oben schieben“ drücken	Kompakt steht oberhalb von Spiegelreflex	Button hat falscher Text statt „Klasse	Not ok

			nach oben schieben“ steht „Knoten nach oben schieben“	
04.17	Button „Klasse nach unten schieben“ drücken	Kompakt steht unterhalb von Spiegelreflex	Button hat falscher Text	Not ok
04.18	Klasse Canon EOS 5D MKIII auf die Spiegelreflex Klasse im Baum ziehen	Canon EOS 5D MKIII wird zwischen Spiegelreflex und Kompakt angezeigt		Ok
04.19	Name von „Canon EOS 5D MKIII“ auf „Canon EOS 5D“ ändern	Name wird in der Attributliste sowie im Baum beim Verlassen des Feldes aktualisiert		Ok
04.20	Produkttyp Definition für Canon EOS 5D anwählen	Das „Klassenicon“ (Blau) wird grün (steht für Produkttyp)		Ok
04.21	Bild für Canon EOS 5D wählen (Bildtyp JPG, Breite 528px, Höhe 398px, Horizontale/Vertikale Auflösung 72 dpi, Bittiefe 24, Grösse 66.8KB)	Beispielbild wird angezeigt Im Ordner ./Data/Images/ wurde ein Bild erstellt mit einer GUID als Name und der Extension „jpg“		Ok
04.22	Das Attribut „Shots“ anwählen	Der Tab wechselt auf die Attribute und das Shot Attribut ist selektiert		Ok
04.23	Shots zu Shot umbenennen	Beim Verlassen des Feldes wird der Name in der Attributliste sowie im Baum aktualisiert		Ok
04.24	Das Attribut „Schwenkbares Display“ auf die Klasse Canon EOS 5D ziehen	Das Attribut wurde verschoben und ist jetzt der Klasse Canon EOS 5D zugewiesen		Ok
04.25	Das Attribut Jahr auf die Klasse Kompakt im Baum ziehen	Wird hinzugefügt		Ok
04.26	Attribut Jahr im Baum anwählen und Beziehung entfernen klicken	Jahr wird nur aus der Klasse Kompakt entfernt		Ok
04.27	Das Attribut Jahr auf die Klasse Kompakt im Baum ziehen	Wird hinzugefügt		Ok
04.28	Klasse Kompakt im Baum markieren und Beziehung entfernen klicken	Klasse Kompakt verschwindet aus dem Baum		Ok
04.29	Klasse Kompakt erneut in den Baum einfügen	Klasse wird eingefügt und das Attribut Jahr wird wieder angezeigt		Ok
04.30	Klasse Kompakt aus der Liste auf Klasse Kompakt im Baum ziehen	Nichts geschieht		Ok

04.31	Klasse Foto Kamera aus der Liste auf Klasse Canon EOS 5D ziehen	Fehlermeldung Zyklische Vererbung	Ok

Vererbungshierarchie: Filter & Markierungen Setup

Nr.	Aktion	Soll	Ist	Status
05.01	Klicke auf „Produkte“ im Baum	Ausgewähltes Element wird blau eingefärbt Themen Tab wird geöffnet und Produkte markiert, Metaattribute werden angezeigt		Ok
05.02	Klicke auf „Kompakt“ im Baum	Ausgewähltes Element wird blau eingefärbt Klassen Tab wird geöffnet und Kompakt markiert, Metaattribute werden angezeigt		Ok
05.03	Klicke auf „Shot“ im Baum	Ausgewähltes Element wird blau eingefärbt Attribute Tab wird geöffnet und Shot markiert, Metaattribute werden angezeigt		Ok
05.04	Klicke auf Jahr in der Attributliste	Jahr in der Attributliste wird blau markiert und im Baum grün		
05.05	Das Attribut Grösse der Klasse Kompakt zuweisen			Ok
05.06	Neues Attribut Groesse erfassen			Ok
05.07	Klasse Canon EOS 7D erfassen			Ok
05.08	Klasse Casio Exilim EX-ZR200			Ok
05.09	Casio Exilim EX-ZR200 der Klasse Kompakt im Baum zuweisen			Ok

Klassen-Filter: Ähnliche Namen

Nr.	Aktion	Soll	Ist	Status
06.01	Alle Filter entfernen			Ok
06.02	Wähle Filter	Canon EOS 7D und Canon EOS 5D werden angezeigt		Ok

Klassen-Filter: Klassen mit X gemeinsamen Attributen

Nr.	Aktion	Soll	Ist	Status
07.01	Alle Filter entfernen			Ok
07.02	Wähle Filter Wähle mindestens 1 gemeinsames Attribut	Foto Kamera und Kompakt werden angezeigt		Ok
07.03	Klicke auf Foto Kamera in der Attribut liste	Foto Kamera wird im Baum grün und Kompakt gelb markiert		Ok

Klassen-Filter: Nicht im Vererbungsbaum (und nicht erbend)

Nr.	Aktion	Soll	Ist	Status
08.01	Alle Filter entfernen			Ok
08.02	Wähle Filter	Canon EOS 7D wird angezeigt		Ok

Klassen-Filter: Nicht Produkttypdefinitionen ohne Kinder

Nr.	Aktion	Soll	Ist	Status
09.01	Alle Filter entfernen			Ok
09.02	Wähle Filter	Canon EOS 7D und Casio Exilim EX-ZR200 werden angezeigt		Ok

Attribut-Filter: Ähnliche Namen

Nr.	Aktion	Soll	Ist	Status
10.01	Alle Filter entfernen			Ok
10.02	Wähle Filter	Groesse und Grösse werden angezeigt		Ok

Attribut-Filter: Mehrfachverwendete Attribute

Nr.	Aktion	Soll	Ist	Status
11.01	Alle Filter entfernen			Ok
11.02	Wähle Filter (Standard 2 Attribute)	Grösse wird angezeigt		Ok
11.03	Grösse in der Attributliste markieren	Grösse wird im Baum zwei Mal grün Markiert		Ok

Attribut-Filter: Nie verwendete Attribute

Nr.	Aktion	Soll	Ist	Status
12.01	Alle Filter entfernen			Ok
12.02	Wähle Filter	Groesse wird angezeigt		Ok

Attribut-Filter: (Semantisch) Name

Nr.	Aktion	Soll	Ist	Status
13.01	Alle Filter entfernen			Ok
13.02	Wähle Filter	Nichts wird angezeigt		Ok
13.03	Setzte Filter: ist / Jahr	Jahr wird angezeigt		Ok
13.04	Setze Filter: beinhaltet / ö	Grösse wird angezeigt		Ok
13.05	Setze Filter: ist leer	Textfeld wird deaktiviert, keine Attribute werden angezeigt		Ok
13.06	Setze Filter: ist nicht leer	Textfeld wird deaktiviert, 9 Attribut werden angezeigt		Ok

Attribut-Filter: (Semantisch) Einheit

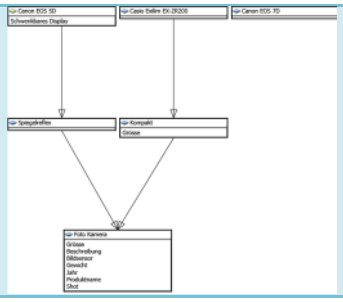
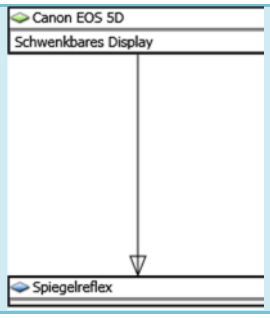
Nr.	Aktion	Soll	Ist	Status
14.01	Alle Filter entfernen			Ok
14.02	Wähle Filter			Ok
14.03	Setzte Filter: ist gleich / Keine	Beschreibung wird angezeigt		Ok
14.04	Setze Filter: ist nicht gleich / Keine	8 Attribute werden angezeigt, Beschreibung nicht		Ok
14.05	Setze Filter: ist leer	6 Attribute werden angezeigt		Ok
14.06	Setze Filter: ist nicht leer	3 Attribute werden angezeigt		Ok

Attribut-Filter: (Technisch) Write-Once

Nr.	Aktion	Soll	Ist	Status
15.01	Alle Filter entfernen			Ok
15.02	Wähle Filter			Ok
15.03	Setzte Filter: ist gesetzt	Kein Attribut wird angezeigt		Ok
15.04	Setze Filter: nicht gesetzt	9 Attribute werden angezeigt		Ok

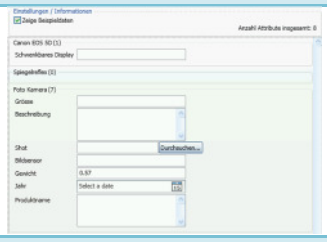
UML-Ansicht

Nr.	Aktion	Soll	Ist	Status
16.01	Generiere UML für alle Klassen	y		Ok

				
16.02	Exportiere UML			Ok
	Wähle Spiegelreflex Klasse und klicke Generiere UML ab ausgewählter Klasse			Ok

Frontend Vorschau

Eingabemasken

Nr.	Aktion	Soll	Ist	Status
17.01	Wähle Canon EOS 5D			Ok
17.02	Wähle Zeige Beispieldaten			Ok
17.03	Maus über Gewicht-Eingabefeld	Tooltip: Gewicht der Kamera erscheint		Ok

Metadaten Strukturierung

Klassen

Nr.	Aktion	Soll	Ist	Status
18.01	Wähle Gruppe Semantisch	Aktivierte Buttons: • Gruppe erstellen • Metaattribute erstellen • Gruppe nach unten schieben Deaktivierte Buttons: • Gruppe löschen • Gruppe nach oben verschieben Anzeige Gruppenmetaattribute		Ok
18.02	Wähle Name	Aktivierte Buttons: • Gruppe erstellen • Metaattribut nach unten verschieben Deaktivierte Buttons: • Gruppe löschen • Metaattribut nach oben verschieben • Metaattribute erstellen Anzeige Metaattribute des Metaattributes Name		Ok
18.03	Verschiebe Name in Gruppe Channel Vorschau	Channel Vorschau Gruppe kann nicht mehr gelöscht werden In der Datenstrukturierung wird das Metattribut in der neuen Gruppe angezeigt		Ok
18.04	Metaattribut Beispiel Bild löschen	Bild wird gelöscht und im Ordner ./Data/Images befinden sich keine Bilder mehr	`SelectionChangedMessageHandler` ➔ Object reference not set to an instance of an object. Objekt wird nicht glöscht, keine Daten gehen verloren	Not Ok
18.05	Bemerkung: Setze “Zeige das Metaattribut in der Liste“	Metaattribut wird in der Liste angezeigt		Ok
18.06	Bemerkung: Setze “Zeige das Metaattribut in der kleinen Liste“	Metaattribut wird in der kleinen Liste angezeigt		Ok
18.07	Bemerkung: Setze “Zeige das	Metaattribut wird in der		Ok

	Metaattribut in der Produkttyp Liste“	Produkttyp Liste angezeigt		
18.08	Erfasse neues Metaattribut „Kosten“ vom Typ Text in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt		Ok
18.09	Erfasse neues Metaattribut „KostenDetails“ vom Typ Mehrzeiliger Text in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt		Ok
18.10	Erfasse neues Metaattribut „Initialkosten vorhanden“ vom Typ Boolean in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt		Ok
18.11	Erfasse neues Metaattribut „Zweites Produkt Bild“ vom Typ Bild in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt		Ok
18.12	Erfasse neue Gruppe mit dem Name „Test“	Gruppe wird erzeugt		Ok
18.13	Lösche Gruppe Organisatorisch	Gruppe wird gelöscht und in den Listen nicht mehr angezeigt	`SelectionChangedMessageHandler` ➔ Object reference not set to an instance of an object. Objekt wird nicht glöscht, keine Daten gehen verloren	Not Ok

Enumerationen

Nr.	Aktion	Soll	Ist	Status
19.01	Lösche Ampere aus der Liste Einheit	Ampere wird nach Bestätigung gelöscht und in den Datenstrukturierungen nicht mehr angezeigt		Ok
19.02	Lösche Megapixel aus der Liste Einheit	Megapixel wird nach Bestätigung gelöscht und in den Datenstrukturierungen nicht mehr angezeigt Die Einheit des Attributes Bildsensor ist leer		Ok
19.03	Lösche CHF aus der Liste Einheit und klicke auf nein	Nichts passiert		Ok
19.04	Lösche CHF aus der Liste Einheit und klicke auf Abbrechen	Nichts passiert		Ok
19.05	Prüfe ob Datentyp nicht in	Nicht vorhanden		Ok

	der Liste der Enumerationen vorhanden ist			
19.06	Wechsel zum Attribut Metaattribute Tab, selektiere Einheit. Wechsle zum Enumerationen Tab und lösche die Einheit	Die Einheit wurde gelöscht und in dem Attribut Metaattribute ist die Einheit auch nicht mehr sichtbar	Die Einheit wurde gelöscht und ist auch nicht mehr in der Liste aber die Metaattribute zu der Einheit werden im Attribut Metaattribute Tab noch angezeigt. Werden die Daten dort jetzt editiert und man wechselt in den Datenstrukturierungs Tab stürzt das Programm ab. Keine Daten gehen verloren und bei einem Neustart arbeitet das Programm weiter.	Not Ok

Konfiguration

Backup

Nr.	Aktion	Soll	Ist	Status
20.01	Erstelle ein Backup	Backup wurde erstellt		Ok
20.02	Lösche das Attribut Grösse	Attribut Grösse gelöscht		Ok
20.03	Lösche die Klasse Canon EOS 5D	Klasse wurde gelöscht		Ok
20.04	Lösche das Thema Produkte	Thema wurde gelöscht		Ok
20.05	Spiele das Backup ein	Alle Daten müssen wieder vorhanden sein		Ok

A.1.1 Import / Export: XML Export

Nr.	Aktion	Soll	Ist	Status
21.01	Exportiere Daten	Fileauswahl-Dialog XML erstellt mit Daten		Ok

A.1.2 Import / Export: Einfacher Attributimport

Nr.	Aktion	Soll	Ist	Status
22.01	Importiere Datei „EinfachAttributListe.txt“	Zwei neue Attribute wurden hinzugefügt Wasserdicht und Zoom		Ok

Import / Export: Attribute und Metaattribute Importer (Tabulator getrennt)

Nr.	Aktion	Soll	Ist	Status
23.01	Importiere Datei „AttributMitMetaattributListe.txt“	Ein neues Attribut Empfindlichkeit mit Beispieldaten Attribut Zoom ist jetzt Internationalisierbar		Ok

Import / Export: Vererbungshierarchie Importer (Tabulator getrennt)

Nr.	Aktion	Soll	Ist	Status
24.01	Importiere Datei „Vererbungshierarchie.txt“	Überprüfe Import		Ok

PSA Konfiguration

Nr.	Aktion	Soll	Ist	Status
25.01	Setze Kundenlogo wähle Beispielbild	Nach Programm Neustart ist das Kundenlogo gesetzt		Ok
25.02	Setze Kundenlogo wähle Beispielbild	Nach Programm Neustart ist das Kundenlogo gesetzt	Absturz: File in Use!	Not Ok

Test Durchführung #2

Setup

Gestartet wird mit einer leeren Datenbank.

Durchführer	Michael Steiner
Datum	30.05.2012
System	Windows XP 32 Bit
.Net Version	4.0.30319

Testresultat Legende

	Test erfolgreich
	Test nicht erfolgreich, Status Unkritisch
	Test nicht erfolgreich, Status Kritisch

Testprotokoll

Daten Strukturierung

Attribute

Nr.	Aktion	Soll	Ist	Status
01.01	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.02	Name „Gewicht“ eingeben	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.03	Beschreibung „Gewicht der Kamera“ eingeben			Ok
01.04	Einheit „Kilogramm“ wählen	Einheit in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.05	Datentyp „Dezimalzahl“ wählen	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.06	Beispieldaten „0.57“			Ok
01.07	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.08	Name „Beschreibung“ eingeben	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.09	Einheit „keine“ wählen	Einheit in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.10	Unterscheidet Variation: Nie	Unterscheidet Variation in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.11	Datentyp „Mehrzeiliger Text“	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.12	Internationalisierbar: ja	Internationalisierbar in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.13	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld		Ok
01.14	Name „Produktname“	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.15	Datentyp „Text mit Platzhalter“	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok
01.16	Unterscheidet Variation: Immer	Unterscheidet Variation in der Tabelle wird		Ok

		aktualisiert beim Verlassen des Feldes	
01.17	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld	Ok
01.18	Name: Bildsensor	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.19	Datentyp: Ganzzahl	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.20	Metadatenstrukturierung → Enumerationen → Einheit → Megapixel erfassen	Megapixel kann der Enumeration hinzugefügt werden und wird angezeigt	Ok
01.21	Zurück zum Attribut Bildsensor und die Einheit „Megapixel“ hinzufügen	Megapixel steht jetzt in dem Dropdown und kann gewählt werden, Einheit in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.22	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld	Ok
01.23	Name: Jahr	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.24	Datentyp: Datum	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.25	Beschreibung: Erscheinungsdatum		Ok
01.26	Auf Suchen Tab wechseln	Tab wechselt	Ok
01.27	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld	Ok
01.28	Name: Shots	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.29	Beschreibung: Beispielbild aufgenommen mit diesem Produkt		Ok
01.30	Datentyp: Binärdaten	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.31	Auf Suchen Tab wechseln	Tab wechselt	Ok
01.32	Filter „Ähnliche Namen anwenden“	Attributliste wird leer	Ok
01.33	Auf Metaattribute Tab wechseln	Tab wechselt	Ok
01.34	Attribut erfassen	Filter wird entfernt, Neues, leeres Attribut Cursor im Namensfeld	Ok

01.35	Name: Schwenkbares Display	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.36	Datentyp: Boolean	Datentyp in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.37	Attribut erfassen	Neues, leeres Attribut Cursor im Namensfeld	Ok
01.38	Name: Grösse	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes	Ok
01.39	Beschreibung: Gibt an ob Spiegelreflex oder Kompaktkamera		Ok

Themen (Daten Strukturierung → Themen)

Nr.	Aktion	Soll	Ist	Status
02.01	Thema erfassen	Neues, leeres Thema Cursor im Namensfeld		Ok
02.02	Name: Produkte	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok

Klassen (Daten Strukturierung → Klassen)

Nr.	Aktion	Soll	Ist	Status
03.01	Button Klasse erstellen muss „Enabled“ sein			Ok
03.02	Button Klasse löschen muss „Disabled sein“			Ok
03.03	Neue Klasse erstellen			Ok
03.04	Name: „Foto Kamera“ erfassen	Name in der Tabelle wird aktualisiert beim Verlassen des Feldes		Ok

Vererbungshierarchie: Drag & Drop

Nr.	Aktion	Soll	Ist	Status
04.02	Auf Vererbungshierarchie Tab wechseln	Folgende Buttons müssen Disabled sein: • Beziehung entfernen • Knoten nach oben schieben • Knoten nach unten schieben Tab „Klassen“ muss geöffnet sein und die Foto Kamera selektiert		Ok

		Der Filter muss leer sein Die Metaattribute der Foto Kamera Klasse müssen geöffnet sein Das Thema „Produkt“ muss im Baum ersichtlich sein	
04.03	Die Klasse Foto Kamera per Drag & Drop auf PSA loslassen	Nichts passiert	Ok
04.04	Die Klasse Foto Kamera per Drag & Drop auf die weisse Fläche in der Baumstruktur ziehen	Nichts passiert	Ok
04.05	Die Klasse Foto Kamera per Drag & Drop auf den Filter ziehen und loslassen	Nichts passiert	Ok
04.06	Die Klasse Foto Kamera per Drag & Drop auf das Thema Produkte ziehen und loslassen	Drop Symbol erscheint und Klasse wird unterhalb des Themas angezeigt	Ok
04.07	In den Attribut Tab wechseln		Ok
04.08	Jedes Attribut auf die Klasse ziehen (Alphabetisch sortiert beginnen, alle ausser Shots)	Nichts passiert	Ok
04.09	Jedes Attribut auf die Attributspalte der Klasse „Foto Kamera“ ziehen (Alphabetisch sortiert beginnen, alle ausser Shots)	Attribute werden der Klasse hinzugefügt	Ok
04.10	Attribut Shots auf das Attribut Beschreibung im Baum ziehen	Shots wird hinter der Beschreibung eingefügt	Ok
04.11	Zum Klassen Tab wechseln		Ok
04.12	Drei Klassen erfassen: Spiegelreflex, Kompakt und Canon EOS 5D MKIII		Ok
04.13	Die Klasse Spiegelreflex auf die Klasse „Foto Kamera“ im Baum ziehen	Die Klasse wird unten angefügt	Ok
04.14	Die Klasse Kompakt auf die Klasse „Spiegelreflex“ im Baum ziehen	Die Klasse wird unten angefügt	Ok
04.15	Die Klasse Kompakt im Baum auf die Klasse Foto Kamera ziehen	Kompakt ist jetzt auf gleicher Höhe (vererbungstiefe) wie Spiegelreflex Kompakt ist markiert	Ok
04.16	Button „Klasse nach oben schieben“ drücken	Kompakt steht oberhalb von Spiegelreflex Kompakt ist markiert	Ok

04.17	Button „Klasse nach unten schieben“ drücken	Kompakt steht unterhalb von Spiegelreflex Kompakt ist markiert	Ok
04.18	Klasse Canon EOS 5D MKIII auf die Spiegelreflex Klasse im Baum ziehen	Canon EOS 5D MKIII wird zwischen Spiegelreflex und Kompakt angezeigt	Ok
04.19	Name von „Canon EOS 5D MKIII“ auf „Canon EOS 5D“ ändern	Name wird in der Attributliste sowie im Baum beim Verlassen des Feldes aktualisiert	Ok
04.20	Produkttyp Definition für Canon EOS 5D anwählen	Das „Klassenicon“ (Blau) wird grün (steht für Produkttyp)	Ok
04.21	Bild für Canon EOS 5D wählen (Bildtyp JPG, Breite 528px, Höhe 398px, Horizontale/Vertikale Auflösung 72 dpi, Bittiefe 24, Grösse 66.8KB)	Beispielbild wird angezeigt Im Ordner ./Data/Images/ wurde ein Bild erstellt mit einer GUID als Name und der Extension „jpg“	Ok
04.22	Das Attribut „Shots“ anwählen	Der Tab wechselt auf die Attribute und das Shot Attribut ist selektiert	Ok
04.23	Shots zu Shot umbenennen	Beim Verlassen des Feldes wird der Name in der Attributliste sowie im Baum aktualisiert	Ok
04.24	Das Attribut „Schwenkbares Display“ auf die Klasse Canon EOS 5D ziehen	Das Attribut wurde verschoben und ist jetzt der Klasse Canon EOS 5D zugewiesen	Ok
04.25	Das Attribut Jahr auf die Klasse Kompakt im Baum ziehen	Wird hinzugefügt	Ok
04.26	Attribut Jahr im Baum anwählen und Beziehung entfernen klicken	Jahr wird nur aus der Klasse Kompakt entfernt	Ok
04.27	Das Attribut Jahr auf die Klasse Kompakt im Baum ziehen	Wird hinzugefügt	Ok
04.28	Klasse Kompakt im Baum markieren und Beziehung entfernen klicken	Klasse Kompakt verschwindet aus dem Baum	Ok
04.29	Klasse Kompakt erneut in den Baum einfügen	Klasse wird eingefügt und das Attribut Jahr wird wieder angezeigt	Ok
04.30	Klasse Kompakt aus der Liste auf Klasse Kompakt im Baum ziehen	Nichts geschieht	Ok
04.31	Klasse Foto Kamera aus der Liste auf Klasse Canon EOS	Fehlermeldung Zyklische Vererbung	Ok

	5D ziehen	

Vererbungshierarchie: Filter & Markierungen Setup

Nr.	Aktion	Soll	Ist	Status
05.01	Klicke auf „Produkte“ im Baum	Ausgewähltes Element wird blau eingefärbt Themen Tab wird geöffnet und Produkte markiert, Metaattribute werden angezeigt		Ok
05.02	Klicke auf „Kompakt“ im Baum	Ausgewähltes Element wird blau eingefärbt Klassen Tab wird geöffnet und Kompakt markiert, Metaattribute werden angezeigt		Ok
05.03	Klicke auf „Shot“ im Baum	Ausgewähltes Element wird blau eingefärbt Attribute Tab wird geöffnet und Shot markiert, Metaattribute werden angezeigt		Ok
05.04	Klicke auf Jahr in der Attributliste	Jahr in der Attributliste wird blau markiert und im Baum grün		
05.05	Das Attribut Grösse der Klasse Kompakt zuweisen			Ok
05.06	Neues Attribut Groesse erfassen			Ok
05.07	Klasse Canon EOS 7D erfassen			Ok
05.08	Klasse Casio Exilim EX-ZR200			Ok
05.09	Casio Exilim EX-ZR200 der Klasse Kompakt im Baum zuweisen			Ok

Klassen-Filter: Ähnliche Namen

Nr.	Aktion	Soll	Ist	Status
06.01	Alle Filter entfernen			Ok
06.02	Wähle Filter	Canon EOS 7D und Canon EOS 5D werden angezeigt		Ok

Klassen-Filter: Klassen mit X gemeinsamen Attributen

Nr.	Aktion	Soll	Ist	Status
07.01	Alle Filter entfernen			Ok

07.02	Wähle Filter Wähle mindestens 1 gemeinsames Attribut	Foto Kamera und Kompakt werden angezeigt	Ok
07.03	Klicke auf Foto Kamera in der Attribut liste	Foto Kamera wird im Baum grün und Kompakt gelb markiert	Ok

Klassen-Filter: Nicht im Vererbungsbaum (und nicht erbend)

Nr.	Aktion	Soll	Ist	Status
08.01	Alle Filter entfernen			Ok
08.02	Wähle Filter	Canon EOS 7D wird angezeigt		Ok

Klassen-Filter: Nicht Produkttypdefinitionen ohne Kinder

Nr.	Aktion	Soll	Ist	Status
09.01	Alle Filter entfernen			Ok
09.02	Wähle Filter	Canon EOS 7D und Casio Exilim EX-ZR200 werden angezeigt		Ok

Attribut-Filter: Ähnliche Namen

Nr.	Aktion	Soll	Ist	Status
10.01	Alle Filter entfernen			Ok
10.02	Wähle Filter	Groesse und Grösse werden angezeigt		Ok

Attribut-Filter: Mehrfachverwendete Attribute

Nr.	Aktion	Soll	Ist	Status
11.01	Alle Filter entfernen			Ok
11.02	Wähle Filter (Standard 2 Attribute)	Grösse wird angezeigt		Ok
11.03	Grösse in der Attributliste markieren	Grösse wird im Baum zwei Mal grün Markiert		Ok

Attribut-Filter: Nie verwendete Attribute

Nr.	Aktion	Soll	Ist	Status
12.01	Alle Filter entfernen			Ok
12.02	Wähle Filter	Groesse wird angezeigt		Ok

Attribut-Filter: (Semantisch) Name

Nr.	Aktion	Soll	Ist	Status
13.01	Alle Filter entfernen			Ok

13.02	Wähle Filter	Nichts wird angezeigt	Ok
13.03	Setzte Filter: ist / Jahr	Jahr wird angezeigt	Ok
13.04	Setze Filter: beinhaltet / ö	Grösse wird angezeigt	Ok
13.05	Setze Filter: ist leer	Textfeld wird deaktiviert, keine Attribute werden angezeigt	Ok
13.06	Setze Filter: ist nicht leer	Textfeld wird deaktiviert, 9 Attribut werden angezeigt	Ok

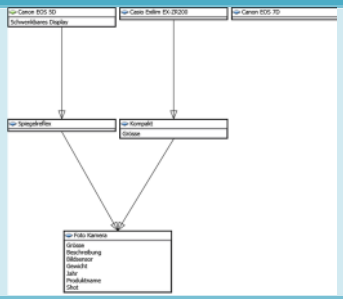
Attribut-Filter: (Semantisch) Einheit

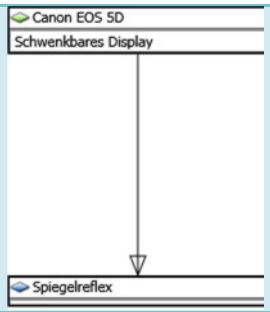
Nr.	Aktion	Soll	Ist	Status
14.01	Alle Filter entfernen			Ok
14.02	Wähle Filter			Ok
14.03	Setzte Filter: ist gleich / Keine	Beschreibung wird angezeigt		Ok
14.04	Setze Filter: ist nicht gleich / Keine	8 Attribute werden angezeigt, Beschreibung nicht		Ok
14.05	Setze Filter: ist leer	6 Attribute werden angezeigt		Ok
14.06	Setze Filter: ist nicht leer	3 Attribute werden angezeigt		Ok

Attribut-Filter: (Technisch) Write-Once

Nr.	Aktion	Soll	Ist	Status
15.01	Alle Filter entfernen			Ok
15.02	Wähle Filter			Ok
15.03	Setzte Filter: ist gesetzt	Kein Attribut wird angezeigt		Ok
15.04	Setze Filter: nicht gesetzt	9 Attribute werden angezeigt		Ok

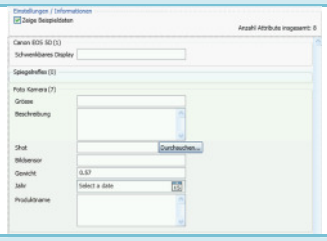
UML-Ansicht

Nr.	Aktion	Soll	Ist	Status
16.01	Generiere UML für alle Klassen			Ok
16.02	Exportiere UML			Ok

	Wähle Spiegelreflex Klasse und klicke Generiere UML ab ausgewählter Klasse			Ok

Frontend Vorschau

Eingabemasken

Nr.	Aktion	Soll	Ist	Status
17.01	Wähle Canon EOS 5D			Ok
17.02	Wähle Zeige Beispieldaten			Ok
17.03	Maus über Gewicht-Eingabefeld	Tooltip: Gewicht der Kamera erscheint		Ok

Christoph Rosenberger
Michael Steiner

Metadaten Strukturierung

Klassen

Nr.	Aktion	Soll	Ist	Status
18.01	Wähle Gruppe Semantisch	Aktivierte Buttons: • Gruppe erstellen • Metaattribute erstellen • Gruppe nach unten schieben Deaktivierte Buttons: • Gruppe löschen • Gruppe nach oben verschieben Anzeige Gruppenmetaattribute		Ok
18.02	Wähle Name	Aktivierte Buttons: • Gruppe erstellen • Metaattribut nach unten verschieben Deaktivierte Buttons: • Gruppe löschen • Metaattribut nach oben verschieben • Metaattribute erstellen Anzeige Metaattribute des Metaattributes Name		Ok
18.03	Verschiebe Name in Gruppe Channel Vorschau	Channel Vorschau Gruppe kann nicht mehr gelöscht werden In der Datenstrukturierung wird das Metattribut in der neuen Gruppe angezeigt		Ok
18.04	Metaattribut Beispiel Bild löschen	Bild wird gelöscht und im Ordner ./Data/Images befinden sich keine Bilder mehr		Ok
18.05	Bemerkung: Setze “Zeige das Metaattribut in der Liste“	Metaattribut wird in der Liste angezeigt		Ok
18.06	Bemerkung: Setze “Zeige das Metaattribut in der kleinen Liste“	Metaattribut wird in der kleinen Liste angezeigt		Ok
18.07	Bemerkung: Setze “Zeige das Metaattribut in der Produkttyp Liste“	Metaattribut wird in der Produkttyp Liste angezeigt		Ok
18.08	Erfasse neues Metaattribut „Kosten“ vom Typ Text in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt		Ok

18.09	Erfasse neues Metaattribut „KostenDetails“ vom Typ Mehrzeiliger Text in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt	Ok
18.10	Erfasse neues Metaattribut „Initialkosten vorhanden“ vom Typ Boolean in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt	Ok
18.11	Erfasse neues Metaattribut „Zweites Produkt Bild“ vom Typ Bild in der Gruppe Semantisch	Metaattribut wird erzeugt und in den Listen angezeigt	Ok
18.12	Erfasse neue Gruppe mit dem Name „Test“	Gruppe wird erzeugt	Ok
18.13	Lösche Gruppe Organisatorisch	Gruppe wird gelöscht und in den Listen nicht mehr angezeigt	Ok

Enumerationen

Nr.	Aktion	Soll	Ist	Status
19.01	Lösche Ampere aus der Liste Einheit	Ampere wird nach Bestätigung gelöscht und in den Datenstrukturierungen nicht mehr angezeigt		Ok
19.02	Lösche Megapixel aus der Liste Einheit	Megapixel wird nach Bestätigung gelöscht und in den Datenstrukturierungen nicht mehr angezeigt Die Einheit des Attributes Bildsensor ist leer		Ok
19.03	Lösche CHF aus der Liste Einheit und klicke auf nein	Nichts passiert		Ok
19.04	Lösche CHF aus der Liste Einheit und klicke auf Abbrechen	Nichts passiert		Ok
19.05	Prüfe ob Datentyp nicht in der Liste der Enumerationen vorhanden ist	Nicht vorhanden		Ok
19.06	Wechsel zum Attribut Metaattribute Tab, selektiere Einheit. Wechsle zum Enumerationen Tab und lösche die Einheit	Die Einheit wurde gelöscht und in dem Attribut Metaattribute ist die Einheit auch nicht mehr sichtbar		Ok

Konfiguration

Backup

Nr.	Aktion	Soll	Ist	Status
20.01	Erstelle ein Backup	Backup wurde erstellt		Ok
20.02	Lösche das Attribut Grösse	Attribut Grösse gelöscht		Ok
20.03	Lösche die Klasse Canon EOS 5D	Klasse wurde gelöscht		Ok
20.04	Lösche das Thema Produkte	Thema wurde gelöscht		Ok
20.05	Spiele das Backup ein	Alle Daten müssen wieder vorhanden sein		Ok

Import / Export: XML Export

Nr.	Aktion	Soll	Ist	Status
21.01	Exportiere Daten	Fileauswahl-Dialog XML erstellt mit Daten		Ok

Import / Export: Einfacher Attributimport

Nr.	Aktion	Soll	Ist	Status
22.01	Importiere Datei „EinfachAttributListe.txt“	Zwei neue Attribute wurden hinzugefügt Wasserdicht und Zoom		Ok

Import / Export: Attribute und Metaattribute Importer (Tabulator getrennt)

Nr.	Aktion	Soll	Ist	Status
23.01	Importiere Datei „AttributMitMetaattributListe.txt“	Ein neues Attribut Empfindlichkeit mit Beispieldaten Attribut Zoom ist jetzt Internationalisierbar		Ok

Import / Export: Vererbungshierarchie Importer (Tabulator getrennt)

Nr.	Aktion	Soll	Ist	Status
24.01	Importiere Datei „Vererbungshierarchie.txt“	Überprüfe Import		Ok

PSA Konfiguration

Nr.	Aktion	Soll	Ist	Status
25.01	Setze Kundenlogo wähle Beispielbild	Nach Programm Neustart ist das Kundenlogo gesetzt		Ok
25.02	Setze Kundenlogo wähle Beispielbild	Nach Programm Neustart ist das Kundenlogo gesetzt		Ok

Anhang E5
Projektverlauf

Projektmanagement Plan

Grundsätzliche Arbeitsweisen

4-Augen-Prinzip

Wir haben verschiedene Verantwortungen auf Rollen aufgeteilt. Die Idee ist aber nicht eine ausschliessliche Verantwortung. Es geht eher darum, dass die jeweilige Person ein besonderes Augenmerk auf ihre Themen hat und allfällige Entscheidungen vorbereitet. Entscheidungen sollten immer von beiden gemeinsam gefällt werden und - nach dem 4-Augen-Prinzip - sollten auch immer beide einen Überblick über die ganze Arbeit haben.

Laufende Dokumentation

Alle Arbeitsschritte werden jeweils dermassen dokumentiert, dass am Ende der Arbeit nur noch die verschiedenen Dokumentationen zu einer End-Dokumentation zusammengefügt werden müssen. Dies bezieht sich sowohl auf den Inhalt, den Umfang und die Qualität.

Dokumentation Projektmanagement

Wir erstellen nicht nur eine Dokumentation über PSA, sondern dokumentieren auch unsere Arbeit anundfürsich. Hierfür erstellen wir Templates z.B. für Sitzungen. Dadurch wollen wir zum einen verhindern, dass wesentliche Aspekte vergessen werden (z.B. beim erfüllen einer Rolle); zum Anderen möchten wir uns so eine Grundlage schaffen, um am Ende des Projekts zu Reflektieren und verstehen, was Gut geloffen ist und was hätte besser laufen können. Um dieses Ziel zu erreichen genügt eine kurzgehaltene bis stichwortartige Dokumentation.

Wöchentliche Team Sitzung

Jede Woche halten wir eine Team Sitzung ab. Hierbei geht es um die Umsetzung der zugeteilten Rollen. Die Sitzungen sollen von den Rollenträgern entsprechend vorbereitet werden, so dass während den Sitzungen gemäss dem 4-Augen-Prinzip effizient die Situation überblickt werden kann und allfällige Entscheide getroffen werden.

Projektorganisation

Wir haben eine sehr offene Aufgabenstellung, es gibt keine Feature Liste. Um dieses Projekt zu bearbeiten drängen sich Agile Entwicklungsmethoden geradezu auf. Wir lassen uns insbesondere von RUP und Scrum inspirieren, werden aber nicht streng nach einem dieser Modelle Vorgehen. Wir werden jeweils in Iterationen das Programm evolutionär entwickeln.

Rollen

Projekt Manager

Person CR

Beschreibung

Der Projektmanager ist verantwortlich für die grundsätzliche Organisation und Planung des Projekts. Im Gegensatz zum Projectowner ist er nicht auf inhaltliche Fragen fokussiert, sondern hat auch die organisatorischen Details unter Kontrolle. Folgende Punkte sollte er im Auge behalten:

- Wo stehen wir? Welche Arbeiten sind gemacht und welche müssen

noch gemacht werden?
- Werden alle Anforderungen an das Projekt erfüllt?
  - Welche Dokumente und Termine werden von der HSR vorgegeben?
  - Wie wird das Miniprojekt aus dem Modul UInt 2 integriert?
- Wie sieht die längere Planung aus? Gibt es Meilensteine und grössere Reviews?
- Wie soll sich das Team organisieren?
- Wie sieht unser Zeitbudget aus?

Project Owner

Person MS

Beschreibung Die Rolle des Project Owner ist von Scrum inspiriert. Er ist dafür verantwortlich, dass das entwickelte Produkt dem Kunden schlussendlich auch einen Nutzen bringt. Er hat einen Überblick über die entwickelten und noch zu entwickelnden Features und sorgt dafür, dass am Ende des Projekts ein für den Kunden fyayc sinnvolles Produkt erstellt wurde. Er hat folgende Aufgaben:
- Kunden Nutzen überwachen
- Iterationen planen. Er hat einen Überblick über den Backlog und kann bei der Iterationsplanung sagen, welche Features eine sinnvolle Ergänzung sind und was diese zeitlich bedeuten.
- Er Plant auch Iterationen im Voraus, so dass allfällig nötige Meetings mit fyayc Mitarbeitern organisiert werden können.
- Er ist für die Kommunikation mit fyayc verantwortlich.

Risiko Manager

Person CR

Beschreibung Hier geht es um die klassische Rolle des Risiko Managers. Er hat jeweils einen Überblick über die grössten und wahrscheinlichsten Risiken, welche den Erfolg des Projekts gefährden. Wenn nötig erarbeitet er Strategien um die Risiken zu mindern.

Problem Manager

Person MS

Beschreibung Die Rolle des Problem Managers ist vom ScrumMaster inspiriert. Er ist dafür verantwortlich, Probleme welche ein effizientes Weiterkommen im Projekt verhindern, zu beseitigen. Ebenfalls ist er für technische Probleme, z.B. mit eingesetzten Tools, verantwortlich.

Qualität Manager

Person CR

Beschreibung Der Qualität Manager behält den Überblick über die Qualität der Workproducts. Insbesondere erstellt er Berichte über die Codequalität unter Zuhilfenahme von Metric Tools.

Zeitplanung

Termine HSR

Folgende Übersicht stammt aus dem Intranet der HSR (https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html)

TERMINE FRÜHJAHRSSEMESTER 2012

Bachelorarbeit	
bis Mitte Nov. 2011	Einbringen eigener Themen gemäss E-Mail Aufforderung.
05.12. - 18.12.11	Die Betreuer erfassen ihre Arbeiten im AVT-Tool.
19.12.11	Die Arbeiten werden unter https://avt.hsr.ch um 9 Uhr veröffentlicht.
bis 15.01.12	Die Studierenden können sich um 1 Arbeit mit Priorität 1, um 3 Arbeiten mit Priorität 2 und um 3 Arbeiten mit Priorität 3 bewerben.
bis 03.02.12	Zuteilung durch den Abteilungsvorsteher
bis 19.02.12	Einrichten der Arbeitsplätze in den Labors
20.02.12	Beginn der Bachelorarbeit, Ausgabe der Aufgabenstellung durch die Betreuer.
Mai 2012	Fotoshooting. Genauere Angaben erteilt die Kommunikationsstlle rechtzeitig.
08.06.12	Die Studierenden geben den Abstract für die Diplomarbeitsbroschüre zur Kontrolle an ihren Betreuer/Examinator frei. Die Studierenden erhalten vorgängig vom Studiengangsekretariat die Aufforderung und die Zugangsdaten zur Online-Erfassung des Abstracts für die Broschüre. Die Studierenden senden per Email das A0-Poster zur Prüfung an ihren Examinator/Betreuer. Vorlagen sowie eine ausführliche Anleitung betreffend Dokumentation stehen unter den allgemeinen Infos Diplom-, Bachelor- und Studienarbeiten zur Verfügung.
13.06.12	Der Betreuer/Examinator gibt das Dokument mit dem korrekten und vollständigen Abstract für die Broschüre zur Weiterverarbeitung an das Studiengangsekretariat frei.
15.06.12	Abgabe des Berichtes an den Betreuer bis 12.00 Uhr. Fertigstellung des A0-Posters bis 12.00 Uhr.
15.06.12	**HSR-Forum, Vorträge und Präsentation der Bachelor- und Diplomarbeiten, 13 bis 18 Uhr**
06.08. - 25.08.12	Mündliche BA-Prüfung
28.09.12	Nachmittag Bachelorfeier und Ausstellung der Bachelorarbeiten

Abbildung 1: Termine HSR

Gesamtübersicht Zeitplanung

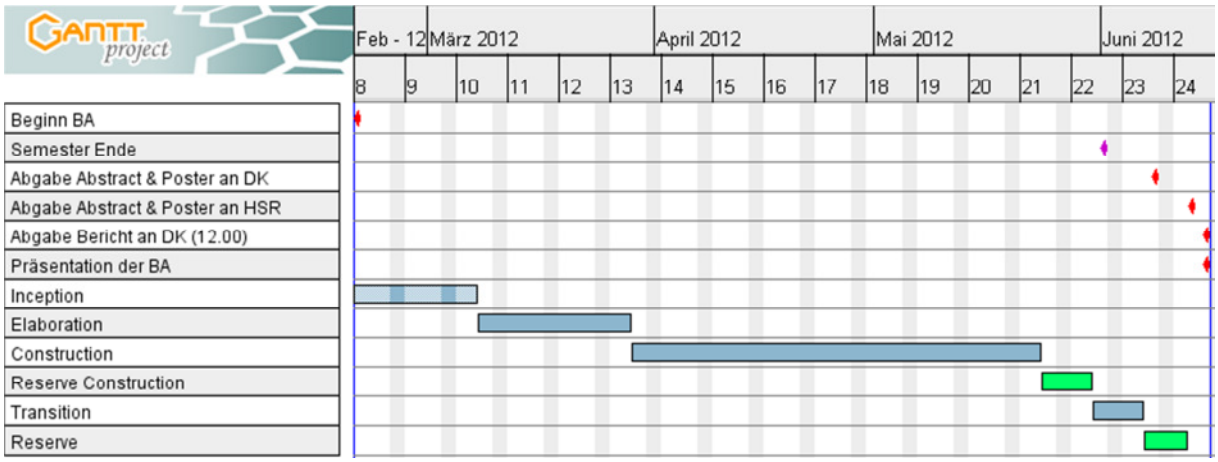


Abbildung 2: Gesamtübersicht Zeitplanung

Zeitbudget

Die Bachelorarbeit wird mit 12 ECTS-Punkten belohnt. Dies entspricht einem Arbeitsaufwand von 12 * 30 = 360 Stunden pro Person. Somit ist unser Zeitbudget insgesamt 720 Stunden.

Arbeitsbelastung

Wir haben 17 Wochen Zeit. 2 Wochen sehen wir als Reserve vor. Die 720 Stunden verteilt auf 15 Wochen und 2 Personen ergibt durchschnittlich 24 Stunden Arbeit pro Person und Woche.

Inception

Zu Beginn des Semesters nehmen wir beide an einem Wettbewerb von Microsoft teil (You Make IT Smart). Ausserdem besuchen wir ein Modul (Business und Recht 2) in welchem wir an einer Business Simulation teilnehmen werden. Dies beinhaltet zu Beginn des Semesters einen grösseren Aufwand für Recherchearbeiten; im Verlauf der Simulation ist der Arbeitsaufwand jedoch geringer.

Daher haben wir uns entschlossen, während der Inception das Arbeitspensum für diese Bachelorarbeit zu reduzieren. Daher dauert die Inception 2.5 Wochen. Die investierten 72 Stunden entsprechen 10% des gesamten Zeitbudgets.

Elaboration

Am Ende der Elaboration wollen wir einen einfachen UML-Editor mit Unterstützung für die Definition und Eingabe von Metaattributen. Somit werden wir nicht nur durch alle Ebenen der Software Architektur gestossen sein, sondern auch die zentralen Entscheide bezüglich einzusetzender Libraries gefällt und das entsprechende Knowhow grossenteils erworben haben.

Wie in der unteren, detaillierteren Planung zu sehen ist, werden wir in dieser Zeit parallel am UInt 2 Miniprojekt arbeiten. Wir planen maximal eine durchschnittliche Arbeitsbelastung; dies ergibt 3 Wochen a 24 Stunden pro Person, also 144 Stunden. Somit wäre die Elaboration maximal 20% des gesamten Projekts.

Construction

Wie beschrieben werden wir zu Beginn der Construction zeitlich im Hintertreffen sein. Dies werden wir durch eine höhere Arbeitsbelastung während dieser Phase wieder ausgleichen.

In diesen 8 Wochen werden wir 456 Stunden Arbeit leisten. Dies entspricht 63% des gesamten Budgets.

Transition

In der Transition werden wir die Dokumentation fertig stellen. Die eigentlichen Inhalte werden wir jeweils zeitnah während des Projekts erstellen. Somit müssen wir während der Transition nur noch die einzelnen Teile zusammen fügen. Ausserdem müssen wir die von der Schule geforderten Abgabeelemente erstellen (Poster, etc.).

Diese Arbeiten müssen in einer Woche bzw. 48 Stunden zu bewältigen sein. Somit wäre die Transition 7% der gesamten Arbeit.

Reserve

Wir haben insgesamt 2 Wochen Reserve eingeplant: Die erste am Ende der Construction, die zweite am Ende des Projekts. Diese Reserve bezieht sich allerdings nur auf die Wochenplanung und nicht auf das Zeitbudget. Sollte dies erforderlich sein, sind wir selbst verständlich bereit auch mehr als die 720 Stunden zu investieren, um unsere Bachelorarbeit erfolgreich abzuschliessen.

Planung Elaboration, Construction und UInt2

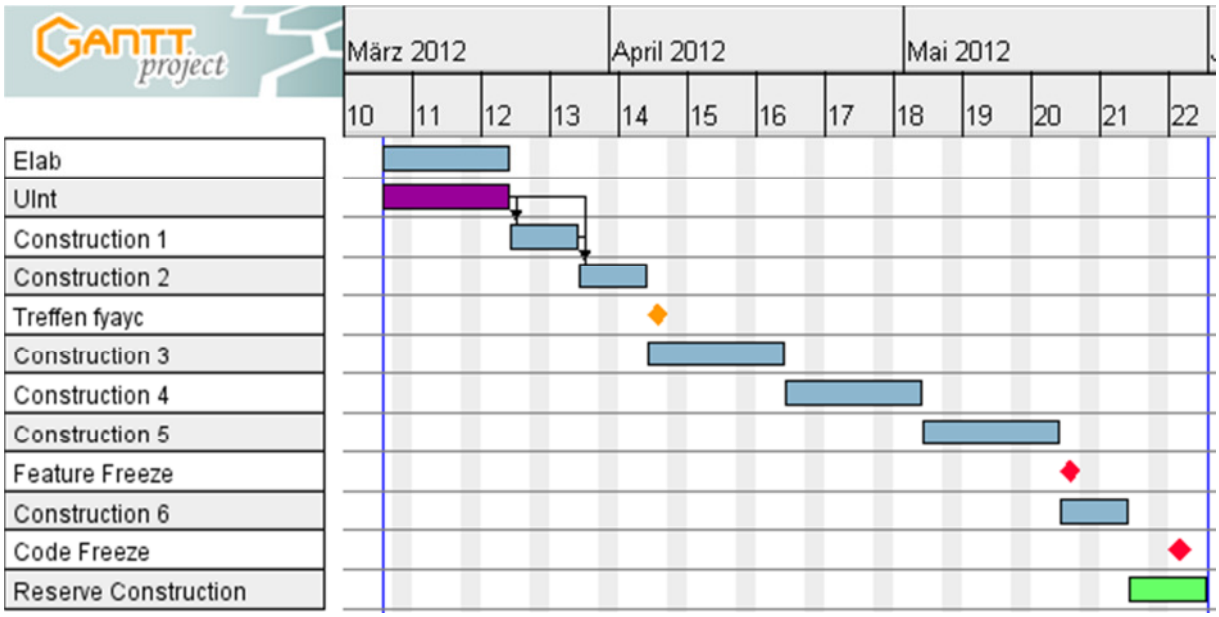


Abbildung 3: Planung Elaboration, Construction und UInt2

Gesamtübersicht

Natürlich ist diese Planung nicht definitiv; die Aufgabenstellung und Zielsetzung ist sehr offen. In dieser Darstellung geht es darum das Zusammenspiel mit dem UInt2 Miniprojekt zu planen und einen Gesamtüberblick über die Construction Phase zu haben.

UInt2 Miniprojekt

Parallel zur Bachelorarbeit besuchen wir das Modul Userinterfaces 2. Somit werden wir Methoden des User Centered Designs kennen lernen. Wir nutzen die Möglichkeit, innerhalb des Miniprojekts zwei Iterationen unserer Bachelorarbeit zu unterstützen. Für das Miniprojekt sind bis zu maximal 30

Stunden Arbeit pro Person gefordert.

Hierfür müssen wir jeweils zusammen mit unserem Auftraggeber , Herrn Keller, ein detailliertes Sollszenario entwerfen, einen Papierprototypen erstellen, diesen Testen und anschliessend überarbeiten.

Hierzu müssen wir folgende Arbeite erledigen:

- 1 kurze Beschreibung wie heute gearbeitet wird.
- 3 detaillierte Sollszenarien entwerfen. Nach einem Feedback von Herrn Keller sollen wir diese dann überarbeiten. Falls nötig sollen wir bis zu drei Feedback-Überarbeitungs-Zyklen durchlaufen.
- Daraus sollen wir 2 Papierprototypen ableiten.
- Die Papierprototypen werden wir mit Herrn Keller testen.
- Anschliessend werden die Papierprototypen Überarbeiten.
- Abschliessend werden wir die Prototypen über alle Garrett Ebenen dokumentieren.

Die so gewonnenen Ergebnisse werden wir in den folgenden Iterationen umsetzen.

Treffen mit fyayc

Die Planung würde es ab der siebten Projektwoche erlauben, ein Treffen mit foryouandyourcustomers zu haben. Ob dies sinnvoll und erwünscht sein wird, werden wir im Verlaufe des Projekts sehen.

2 Wöchige Iterationen

In dieser Übersicht haben wir 2 Wöchige Iterationen eingesetzt. Da wir noch nicht wissen, welche Features wir implementieren werden, ist es auch nicht sinnvoll die Iterationslängen in Stein zu meisseln. Je nach Umfang und Komplexität der Features einer Iteration sind auch einwöchige Iterationen denkbar.

Feature Freeze

Die Anforderung an unsere Software ist eher ein Proof Of Concept um Ideen zu zeigen, als eine Ready To Use Lösung welche produktiv eingesetzt werden soll. Trotzdem erachten wir es für wichtig, dass die umgesetzten Features auch funktionieren. Daher sehen wir für die letzte Woche der Construction einen Feature Freeze vor; d.h. wir werden die letzte Woche keine neuen Features mehr entwickeln, sondern nur noch Bugs beheben.

Code Freeze

Falls am Ende der regulären Construction noch Bugs bekannt sind, haben wir noch eine Woche Reserve. Hier sehen wir spätestens am Dienstag dieser Woche einen Code Freeze vor. Somit hätten wir nötigenfalls noch 3 Tage Zeit, unser Wissen über noch vorhandene Bugs zu dokumentieren.

Keine Reviews

Wir haben bewusst keine Reviews geplant. Da wir mit Herrn Keller jeweils am Ende der Iterationen eine Sitzung haben wo wir auf das Resultat eingehen werden, halten wir spezielle Reviews im Moment nicht für nötig. Je nach Entwicklung des Projekts können wir dies natürlich noch ändern.

Wochenplanung

Sitzung mit Herrn Keller

Jeweils am Donnerstagmorgen findet eine Sitzung mit Herrn Keller statt. Hierbei nimmt er sowohl die Rolle des Dozenten ein als auch diejenige des Kunden bzw. Users von PSA.

Projektmanagement

Die Arbeiten der uns zugeteilten Rollen müssen auf den Donnerstagmorgen vorbereitet werden. Anschliessend an die Sitzung mit Herrn Keller halten wir dann unsere wöchentliche Teamsitzung; diese ist also primär als Projektmanagement Sitzung gedacht.

Iterationen

Unsere Iterationen beginnen und enden ebenfalls am Donnerstag. Dies gibt uns die Möglichkeit mit Herrn Keller die endende Iteration nochmals zu besprechen - sowohl im Sinne einer Reflexion mit Unterstützung des Dozenten, als auch eine Endbesprechung mit dem Kunden um sicher zu stellen, dass wir seine Wünsche und Vorstellungen verstanden und korrekt umgesetzt haben.

Ausserdem können wir so die nächste Iteration Planen und von unserem Kunden entsprechende Details erfragen.

Kick Off Meeting 22.02.2012

Allgemeines

Anwesende cr, ms, dk

Dauer 2h

Traktanden

Fragen

Fragen	Antworten

Budget fyayc
• Wieviel Zeit haben wir bei fyayc zugute?
• Welche Ansprechpersonen haben wir?
  ○ Und welches sind ihre Spezialgebiete?
• Welche Arbeiten werden von fyayc erledigt?
  ○ Welche Inputs können wir erwarten?
  ○ Welche Verantwortung nehmt ihr wahr? (Wir sind von euch abhängig...)
• "Meetingplan" ok?
  ○ Pro Iteration: Input & Kontrolle
  ○ Pro UInt Iteration: Input & Prototyp test & Kontrolle Implementation
  ○ 2 Reviews (stimmt das grosse ganze, wo solls hingehen) (nach z.B. 5 und 11 wochen)
	• Wir können bei fyayc in den Büros arbeiten (wenn Arbeitsplätze frei sind). So könnten wir in den Pausen an den Diskussionen teilnehmen und könnten den Anwesenden Fragen stellen, wenn diese Zeit haben.
	• Wir können schauen, ob jemand Zeit hat und Termine abmachen.

Ziel BA (aus sicht fyayc)
• Was soll am Schluss des Projekts stehen? Was ist das Ziel?
  ○ Ein laufendes Programm? Was soll dieses können?
  ○ Proof of Concept
  ○ Hilfe bei Interner Kommunikation?
  ○ Nichts? (schön für Studis, dass sie eine Arbeit machen können...)
  ○ wie wollt ihr das einsetzen?
• PSA
  ○ Welche Features müssen umgesetzt sein (funktional)
  ▪ Was darstellen
  ▪ Navigation
  ▪ Editieren
  ○ Nicht funktionale Anforderungen
• Wie sehen die Einsatz Szenarios aus?
  ○ Wie werdet ihr das Einsetzen?
  ○ Welchen Zweck soll PSA erfüllen?
	• Am ehsten Proof of Concept. Es darum gute Ideen aufzuzeigen.
	• Ziel ist Hilfe bei interner Kommunikation und mit dem Kunden.
	• Ziel ist Förderung des Einsatzes von UML Klassendiagrammen innerhalb fyayc.

Christoph Rosenberger
Michael Steiner

- Welche Probleme soll PSA lösen? - Welchen Mehrwert bringt es euch? - Was sind eure Erwartungen? - Welche Inputs können wir von fyayc erwarten? - zuverlässig - in welcher Qualität - Welche Inputs können wir von fyayc fordern?	
Hr. Keller als Dozent - Was erwarten Sie von dieser BA? - Wieviel Zeit haben wir zugute? - Welche Inputs können wir erwarten? - Welche Inputs können wir fordern?	- Am Donnerstag von 8.00 bis 10.00 haben wir standardmässig ein Treffen - Wir können natürlich auch weitere Treffen arrangieren.
Aufgabenstellung - Scope von BA - Requirements	

Was wir brauchen

Fragen	Antworten
- ein *reales* Beispiel - reale Mengen (anz. Klassen, etc.) - ansonsten können wir weder sinnvolle Darstellungen planen, noch sehen, ob unser Tool den Zweck erfüllt - Datenmodell	Wir haben ein (vertrauliches) Beispiel einer PIM Schnittstelle bekommen.

Messages an DK

Fragen	Antworten
- vieles ist Unklar - ich fühle mich mit Aufgabenstellung nicht besonders wohl - WIR SIND ABHÄNGIG VON FYAYC - wir werden viel Zeit für Projektmanagement und Risikomanagement brauchen - zu grosser Teil ist immernoch requirements analyse, wengi Software Engineering	- Die Aufgabenstellung wird so interpretiert und verstanden, dass wir eher unabhängig sind von fyayc. - Wir konzentrieren uns auf UML Klassendiagramm und wie man dies präsentiert, wie man Probleme (z.B. Performanceprobleme) erkennen und darstellen kann und wie man mit nicht Informatikern über das Datenmodell sprechen kann.

Organisatorisches

Fragen	Antworten
- Wann Treffen mit DK - fixes Datum möglich? - Wann Treffen mit fyayc - Termin erstes Treffen bei fyayc	- Treffen Donnerstag 8.00 bis 10.00 - Treffen fyayc nicht mehr unmittelbar notwendig. Kann bei Bedarf vereinbart werden.

Neuer Input

- Wie erklärt man, dass Joins (auch im KD) Performance Probleme geben? Wenn man davon ausgeht, dass man nicht weiss, was ein Join ist?
- PIM Data Export Datenmodell V3
- Aufzeigen wie viele SQL-Statements generiert werden müssen für Lösung A und für eine Lösung B
- Verbindungen im UML welche Performance-schwächend sind rot markieren
- Es ist schon ein Fortschritt wenn UML benutzt wird
- Klassen ausblenden, Attribute ausblenden, zoomen
- UML präsentierbar machen

Neue Termine

Standard-Treffen am Donnerstag von 08.10 – 10.10 Uhr.

Für ein Treffen am kommenden Dienstag DK kontaktieren

Sitzung 28.02.2012

Allgemeines

Anwesende cr, ms, dk

Dauer 1h50min

Fragen

Frage	Antwort
Wie wird heute gearbeitet?	• Mit Masken -> Workflow getrieben (Prozess, Logistik) • GUIs -> User Experience, d.h. wie sollen GUI für den Enduser aussehen? • Umsysteme -> IT-Infrastruktur Sicht. Also Diskussion, wie die Daten in den heutigen Systemen abgelegt sind.
Wäre es ein sinnvoller Ansatz, wenn unser Tool beim Erstellen von UML Klassendiagrammen hilft (z.B. einblenden von Erklärungen, was ein bestimmtes Element bedeutet)	Nein, eher nicht. Wir können davon ausgehen, dass die User UML Klassendiagramme verstehen. Kein Schulungstool erstellen.

Neuer Input

- Reale UML sind zu gross für den Beamer (Informationsverdichtung)
- Wie sucht der Konsument Produkte? Kategorie Baum? Faceted Search?
  - Kriterien, Attribute aus dem Datenmodell nehmen und ein Beispiel-Gui generieren.
  - Was ist besser geeignet für ein konkretes Datenmodell?
- Ansichten generieren
- Attribute erfassen aus der Sicht eines Fotografen?
- Se2 Übung 1 → Redmine

Sitzung 01.03.2012

Allgemeines

Anwesende cr, ms, dk

Dauer 2h

Thema

- Besprechung SA

Neuer Input

- Semantische Sachen von der Darstellung trennen (Metadaten) → Wie HTML / CSS
- Metaattribute Kategorisieren, aber mit Mass (max. 5 Kategorien, min. 2)
- Bei Boolschen Werten Aufpassen: Häufig sind mehr als drei Werte wünschenswert. Z.B. Ist ein Attribut nicht internationalisierbar weil es nicht nötig ist oder weil es nicht machbar ist?
- Lieber viele gute Visualisierungen anstelle von Automatisierungen. Lieber fünf nicht automatisierbare Ideen anstatt eine komplett ausprogrammiert und automatisiert

Sitzung 08.03.2012

Allgemeines

Anwesende cr, ms, dk

Dauer 2h

Fragen

Frage	Antwort
Haben Sie Ahnung von Canvas, MVVM, etc. ?	Nein, Luc Bläser fragen
Kennen Sie Metrics Tools für C#?	Nein, Luc Bläser fragen

Weitere Traktanden

- Erklären UInt
  - 2 Szenarien
  - 1. werden wir in einer Woche zusammen entwickeln
  - Detailliert (mit realen Daten)
- Redmine Server beantragen
- Aktueller Zeitaufwand

Neuer Input

Wir haben eine neue Aufgabe gefasst und werden nicht wie geplant fortfahren.

Herr Keller war mit Herr Möller bei einem neuen Kunden und schilderte uns aktuelle Probleme, für welche sie gerne eine bessere Lösung hätten. Somit verschieben wir unsere Pläne auf einen späteren, unbestimmten Zeitpunkt und widmen uns dem neuen Thema.

Es gibt drei neue Teilbereiche:

Attributfindung	Wie können wir die Attribute aller Produkte sauber und strukturiert erfassen? Es sollte in einem möglichst logischen Modell geschehen, so dass auch der Kunde solche Attribute erfassen kann. Was ist mit dem Attributwert? Wie geht man mit einem Attribut um, welches mehrere Attributwerte besitzt (z.B. Breite x Höhe)? Welche Standard-Metaattribute gibt es? Was geschieht mit einer Aktion? Aktionspreis?
Attributvererbungshierarchie	Wie kann man eine Produktkategorisierung / Klassifizierung möglichst logisch und bequem erstellen? Wie ist der Zusammenhang mit der Attributfindung?

Navigationshierarchie	Wie findet man den besten Navigationsbaum? Welche Navigationsbäume gibt es überhaupt / sind möglich?

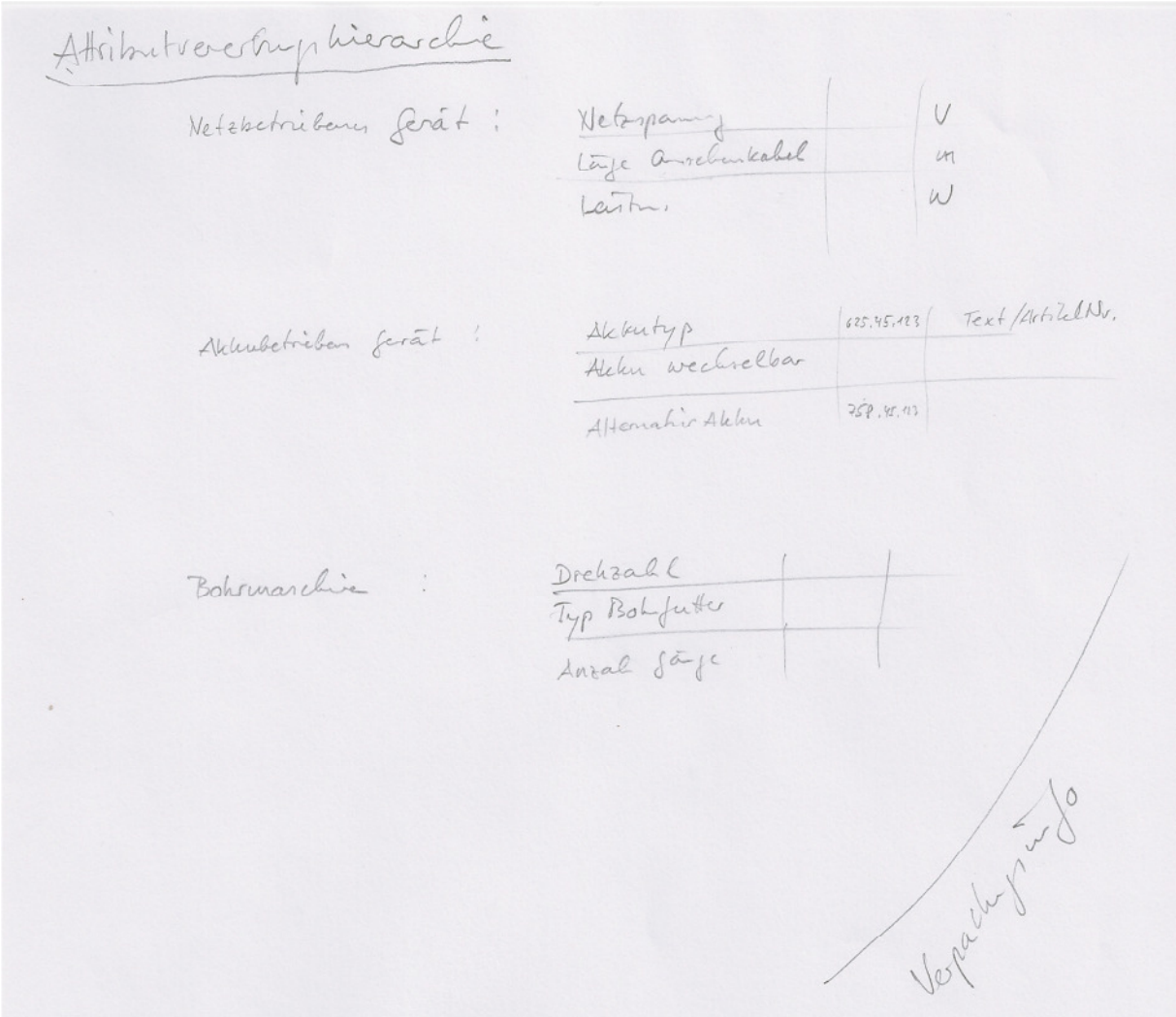
Neue Termine

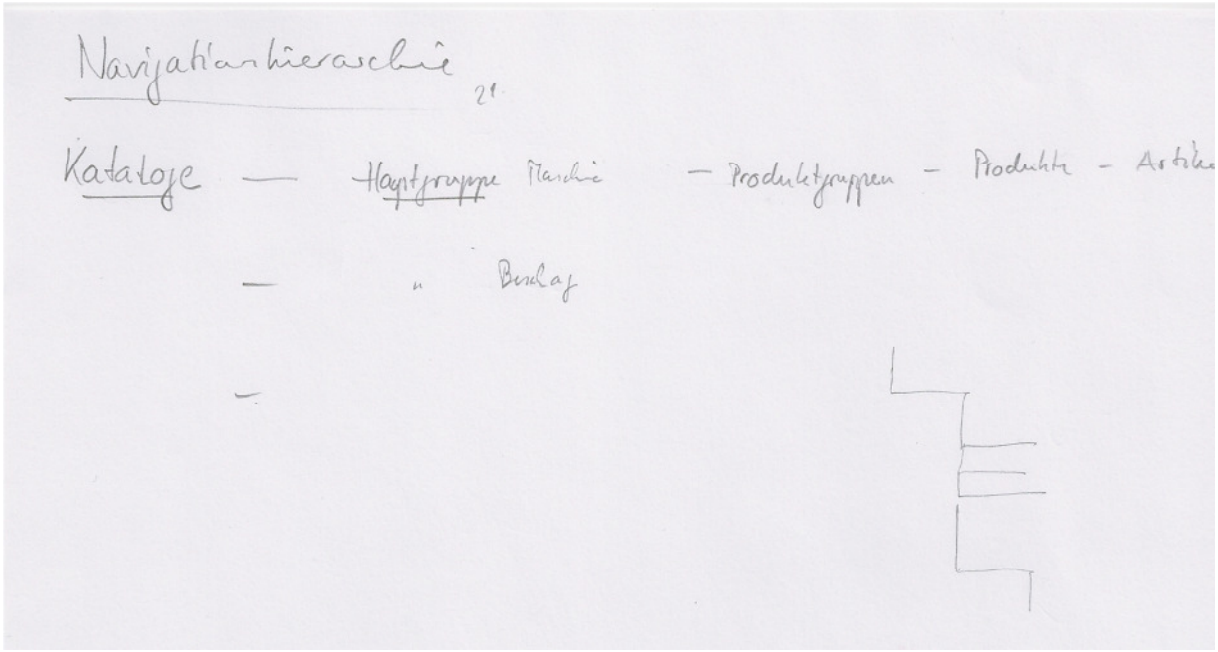
13.3 oder 15.3 per Mail abklären
Herr Rehmann betreffend Redmine kontaktieren
UInt muss neu geplant werden und mit Herr Stolze besprochen werden

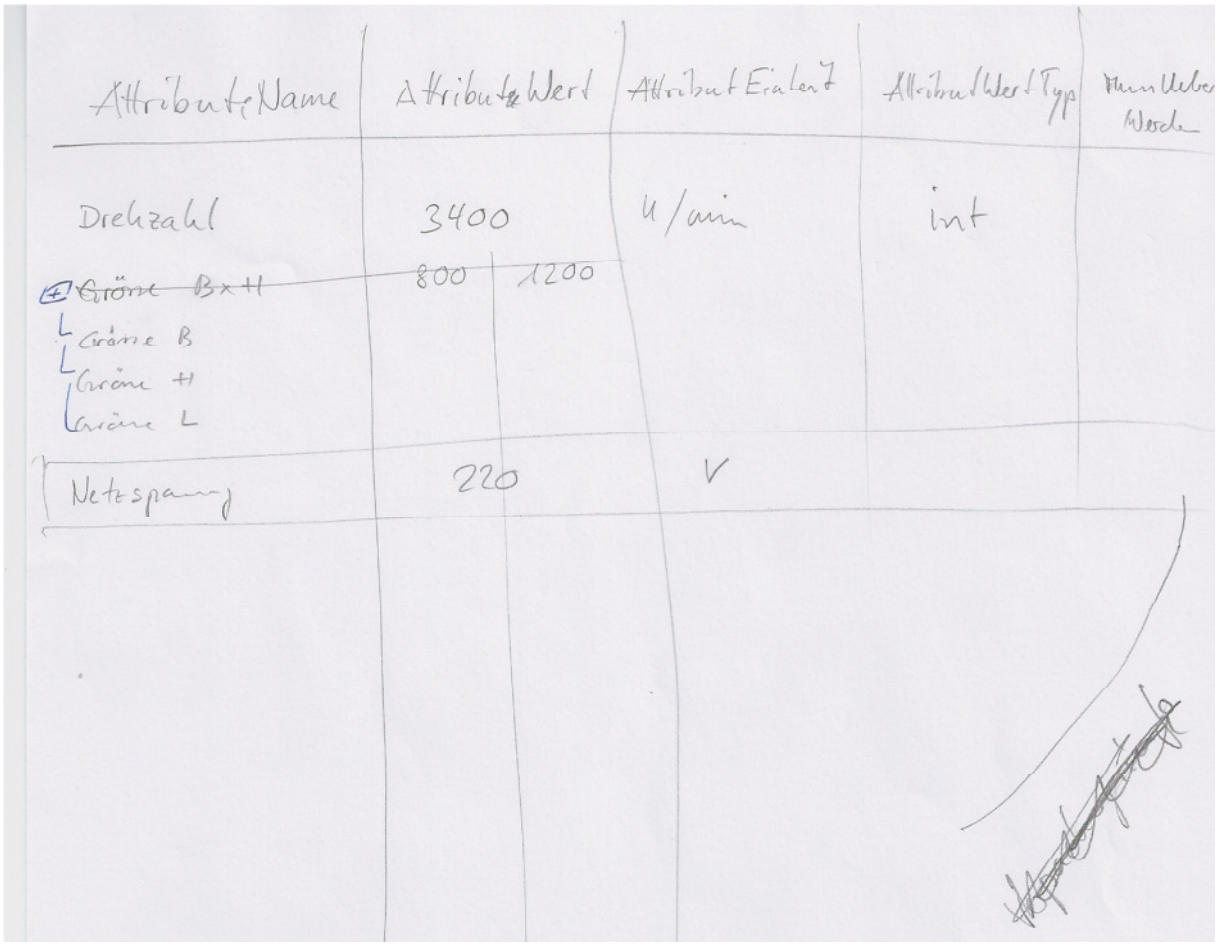
Bemerkungen

Michael: Ich bestelle Produkt A und bekomme einen Vorschlag was ich noch kaufen könnte, ohne dass ich mehr Porto zahlen muss.

Arbeitsblätter




Navigationshierarchie
Kataloge
Hauptgruppe
Produktgruppen
Produkte
Beilage


Attribute Name
Attribute Wert
Attribut Einheit
Attribut Wert Typ
Drehzahl
3400
U/min
int
Größe B x H
800
1200
Größe B
Größe H
Größe L
Netzspannung
220
V

End Of Inception Freitag, 15. Juni 2012

Allgemeines

Projektwoche	3
Anwesende	cr, ms
Dauer	5min

Rollen

Project Owner

Siehe Teamsitzung

Projekt Manager

Zeitauswertung

Wie in der Projektplanung festgehalten, wollten wir bis gestern Donnerstag die Inception abschliessen. Wir hatten 74 Stunden Arbeit geplant.

Durch wesentlichen, aber sehr spannenden neuen Input von Herrn Keller haben wir End Of Inception auf heute Freitagmorgen verschoben.

Wir haben bis jetzt 80 Stunden verbraucht.

Offene Punkte

Folgende Arbeiten müssen über das Weekend noch gemacht werden:

- Redmine einrichten (Wir haben uns erst relativ spät für Redmine entschieden und mussten dann noch ein bisschen auf den Server warten.)
- Einige Dokumente entsprechen noch nicht der Dokumentvorlage.
- Nach dem neuen Input von Herrn Keller müssen wir einen kleinen Teil der Doku anpassen (Zeitplanung, Ausgangslage, Ziele)

Gesamtübersicht

Wir sind im Rahmen unserer Planung.

Teamsitzung Freitag, 15. Juni 2012

Allgemeines

Projektwoche	3
Anwesende	cr, ms
Dauer	20min

Rollen

Project Owner

Kunden-Nutzen vorhanden?	Durch die neue Aufgabenstellung ist der Kundennutzen noch ein bisschen gestiegen.
Backlog sinnvoll abgearbeitet?	-
Was muss mit fyayc kommuniziert werden?	-
Meetings?	Siehe 20120308_Sitzung_DK.docx

Projekt Manager

Zeitauswertung	Geplant war das Ende der Inception für gestern Donnerstagmorgen. Dank neuem Input von Herrn Keller haben wir dies auf heute Morgen verschoben (gleich nach dieser Sitzung werden wir diesen Meilenstein erreichen und würdigen). Es waren 74 Stunden geplant für die Inception, wir haben bis jetzt 77 Stunden verbraucht.
Gesamtübersicht	Wir sind entsprechend unserer Planung unterwegs. Alles im grünen Bereich.

Risiko Manager

Technologie

Beschreibung	Wir haben beide das Modul Microsoft Technologien besucht. Im Rahmen dieses Moduls haben wir sowohl die Programmiersprache C# als auch WPF und das MVVM Pattern kennen gelernt. Im Rahmen eines Microsoft Wettbewerbs (You Make IT Smart) hat

insbesondere Christoph Rosenberger ein wenig Praxiserfahrung in der Programmierung von Silverlight für Windows Phone erhalten.

Somit ist unser Knowhow bezüglich der Basistechnologie eher eingeschränkt.

Mögliche Folgen

- Bugs
- uneinheitlicher Code
- wenig Effizienz zu Beginn des Projekts

Massnahmen

- Insbesondere am Anfang der Entwicklung zu zweit (Pairprogramming).
- Automatisierte Tests um Refactoring zu erleichtern.

UI & Canvas

Beschreibung

Wir haben keinerlei Erfahrung, wie man mit der Canvas umgeht und ob es für unsere Zwecke passende zusätzliche Libraries gibt. Ebenfalls wissen wir nicht, wie eine gute Software Architektur der UI nahen Layer aussehen sollte, ob sich MVVM auch für diesen Aspekt eignet.

Mögliche Folgen

- Zusätzlicher lern Aufwand um mit der Umsetzung zu beginnen.
- Schlechtes, unflexibles Software Design
- Grössere Refactorings nötig

Massnahmen

- Entwicklung des UML Klassendiagramm Editors so früh wie möglich und sinnvoll. (Wird mit einem Papierprototyp geplant, damit wir zielsicher arbeiten können)
- Internet-Recherche Arbeitspaket

Keine Überraschenden Ideen

Beschreibung

Ein wichtiger Teil unserer Arbeit ist die Entwicklung von Ideen. Was aber, wenn uns keine coolen, überraschenden oder wenigstens überzeugenden Ideen und Features in den Sinn kommen?

Mögliche Folgen

- Tool ist langweilig und findet bei potenziellen Usern keinen Anklang.
- Schlechte Note

Massnahmen

- Brainstorming
- Treffen mit fyayc

Problem Manager

Probleme? (Was erschwert uns die Arbeit?)

- Redmine Server läuft noch nicht

Team Reflexion

Arbeitsteilung klappt gut! Motivation ist nach gestrigem Input von DK weiter gestiegen.

Sitzung 13.03.2012

Allgemeines

Anwesende cr, ms, dk

Dauer 1.5h

Thema

- Besprechung Uint-Iteration 1

Sitzung 13.03.2012

Allgemeines

Anwesende cr, ms, dk

Dauer 0.5h

Thema

- Besprechung Uint-Iteration 2

Sitzung 15.03.2012

Allgemeines

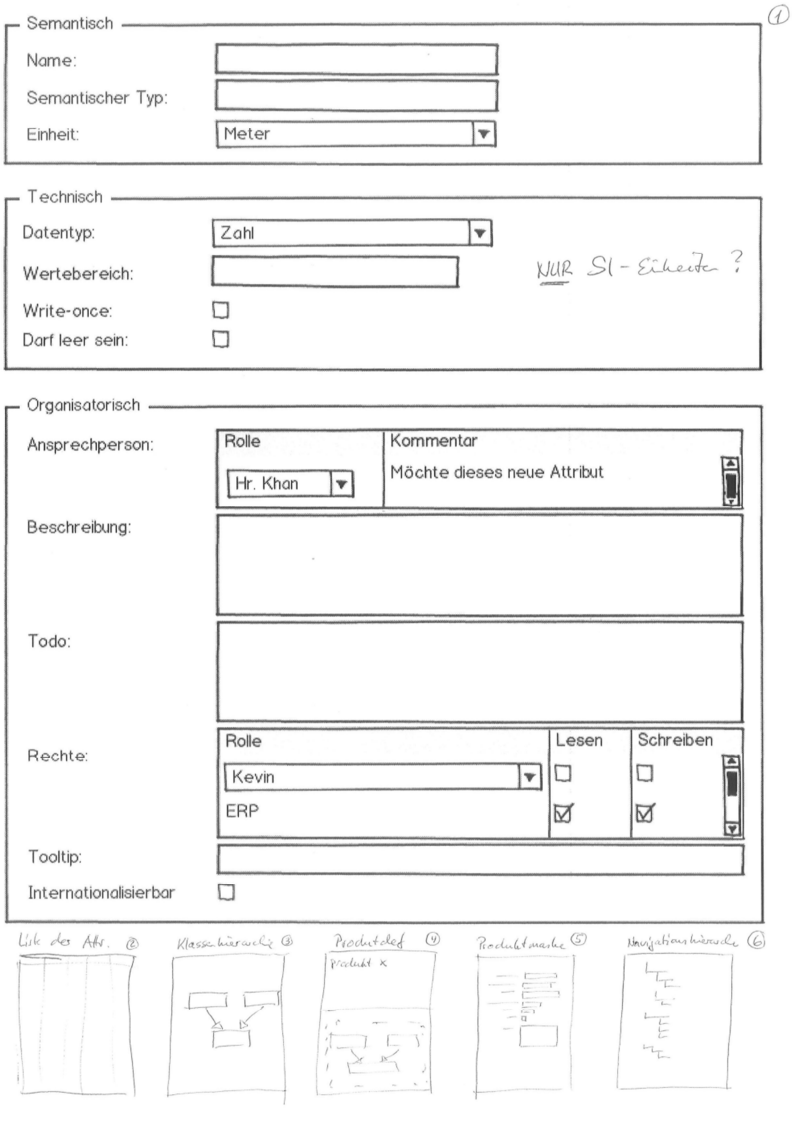
Anwesende cr, ms, dk

Dauer 2h

Thema

- Feature Liste mit fünf Ansichten → würde eine erste lauffähige Variante geben, welche fyayc einsetzen könnte
- Treffen mit fyayc in Woche 7: bis dann werden wir eine erste lauffähige Version haben

- Metaattribute anhand Wireframe diskutiert



Christoph Rosenberger
Michael Steiner

Start Of Elaboration 20.03.2012

Allgemeines

Iteration	Elaboration
Projektwochen	5 bis 7
Datum	20.3.2012 bis 05.04.2012
Dauer (soll)	117.5 Stunden

Einleitung

Ursprüngliche Planung

Die Ursprüngliche Planung hätte vorgesehen, dass wir bis Projektwoche 6 am Donnerstag die Elaboration abschliessen. Inhaltlich hätten wir einen einfachen UML Editor mit Unterstützung für die Definition und Eingabe von Metaattributen entwickelt. Parallel dazu hätten wir das UInt Miniprojekt abgewickelt.

Änderung

Wir haben unsere Planung nun umgestellt. Für unser UInt Miniprojekt haben wir vier Treffen mit Herrn Keller gebraucht. Dieser hat aber in der Projektwoche 5 Ferien. Damit wir möglichst viel Nutzen aus dem UInt Miniprojekt ziehen können und uns dieses nicht behindert, haben wir beschlossen, dieses in Projektwoche 4 abzuwickeln.

Neue Planung

Inhaltlich möchten wir in der Elaboration nun die im UInt Miniprojekt entwickelten Prototypen umsetzen. Zusammen mit den Recherche Arbeiten hat unsere Planung ergeben, dass wir 2.5 Wochen dafür benötigen werden.

Inhaltlich hat sich nichts geändert, wir werden in der 7 Projektwoche ein erstes Release haben. Ein erstes sinnvolles Treffen mit fyayc ist somit auch wie ursprünglich geplant in Projektwoche 7 möglich.

Christoph Rosenberger
Michael Steiner

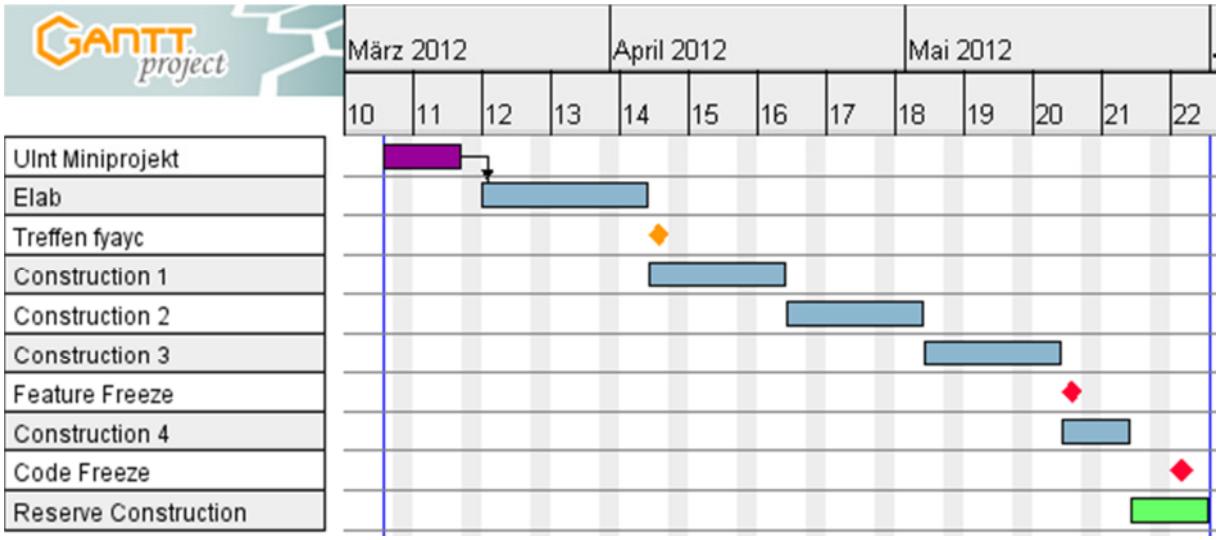


Abbildung 4: Planung UInt, Elab und Construction

Detailplanung

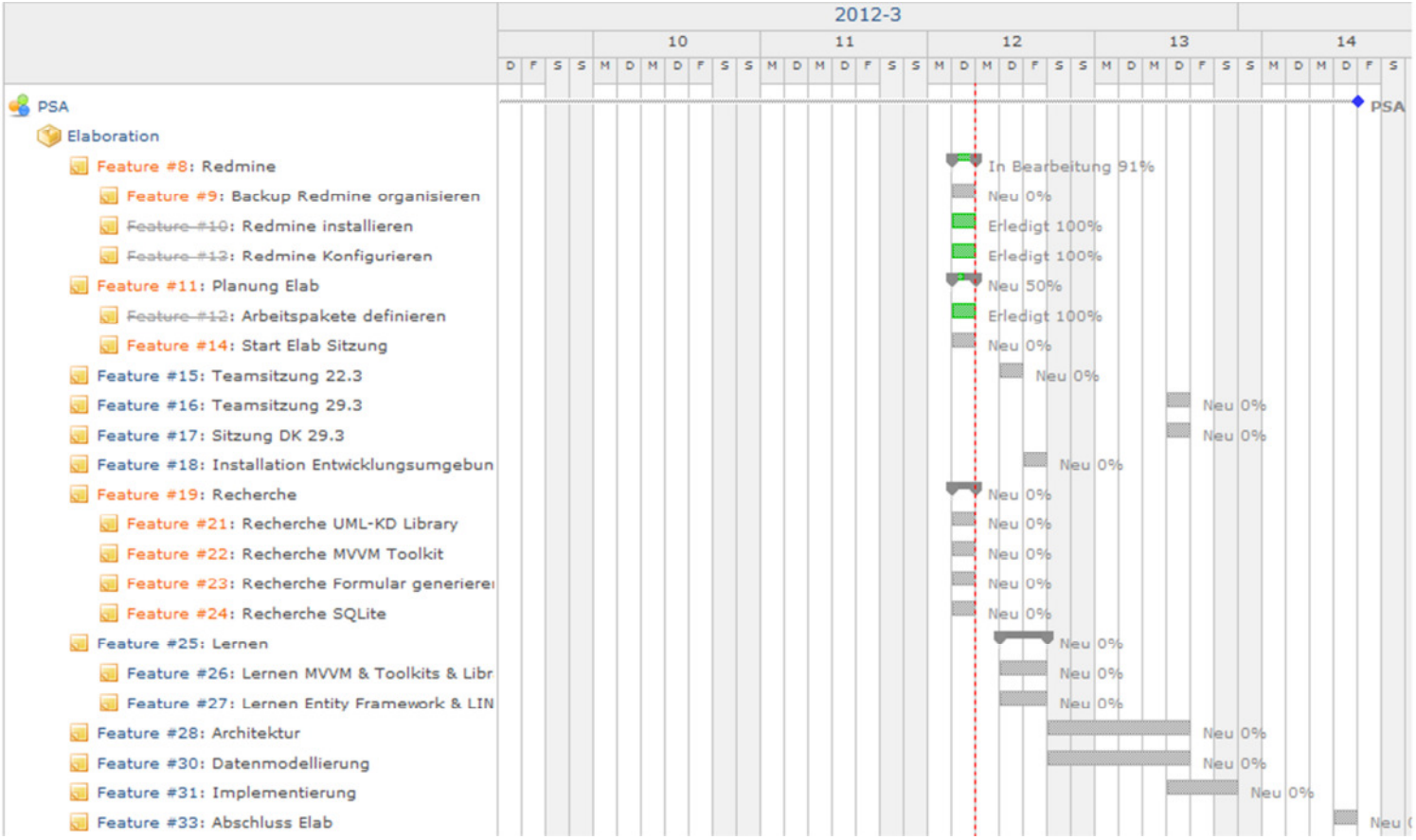


Abbildung 5: Detailplanung Elab (Screenshot Redmine)

Rollen

Projekt Manager

Gesamt Zeit

In der Inception haben wir 80 Stunden gearbeitet. Wir sind nun in Woche 5. Linear gerechnet sollten wir also 192 Stunden gearbeitet haben. Somit sind wir 112 Stunden, also 7 Arbeitstage, im Hintertreffen. Dies entspricht allerdings alles in allem nachwievor unserer Planung.

Geplante Zeit Elab

Für unsere Arbeitspakete haben wir 117.5 Stunden gerechnet. Geteilt durch 7 Arbeitstage und 2 Personen ergibt dies 8.4 Stunden Arbeit pro Arbeitstag.

Um unsere 720 Stunden leisten zu können, sollten wir auch an den Wochenenden Arbeiten. Damit müsste es uns möglich sein, schon früher alle geplanten Arbeiten abzuschliessen. Da aber (naturgemäss) in dieser Elab-Iteration sehr viele Unsicherheiten vorhanden sind, ist diese Reserve durchaus sinnvoll.

Risiko Manager

Technologie (gleich wie Teamsitzung 09.03.2012)

Beschreibung

Wir haben beide das Modul Microsoft Technologien besucht. Im Rahmen dieses Moduls haben wir sowohl die Programmiersprache C# als auch WPF und das MVVM Pattern kennen gelernt.

Im Rahmen eines Microsoft Wettbewerbs (You Make IT Smart) hat insbesondere Christoph Rosenberger ein wenig Praxiserfahrung in der Programmierung von Silverlight für Windows Phone erhalten.

Somit ist unser Knowhow bezüglich der Basistechnologie eher eingeschränkt.

Mögliche Folgen

- Bugs
- uneinheitlicher Code
- wenig Effizienz zu Beginn des Projekts

Massnahmen

- Insbesondere am Anfang der Entwicklung zu zweit (Pairprogramming).
- Automatisierte Tests um Refactoring zu erleichtern.

UI & Canvas (gleich wie Teamsitzung 09.03.2012)

Beschreibung

Wir haben keinerlei Erfahrung, wie man mit der Canvas umgeht und ob es für unsere Zwecke passende zusätzliche Libraries gibt. Ebenfalls wissen wir nicht, wie eine gute Software Architektur der UI nahen Layer aussehen sollte, ob sich MVVM auch für diesen Aspekt eignet.

Mögliche Folgen

- Zusätzlicher lern Aufwand um mit der Umsetzung zu beginnen.
- Schlechtes, unflexibles Software Design

- Grössere Refactorings nötig

Massnahmen

- Entwicklung des UML Klassendiagramm Editors so früh wie möglich und sinnvoll. (Wird mit einem Papierprototyp geplant, damit wir zielsicher arbeiten können)
- Internet-Recherche Arbeitspaket

Sitzung 29.03.2012

Allgemeines

Projektwoche 6

Anwesende cr, ms, dk

Dauer 1:45

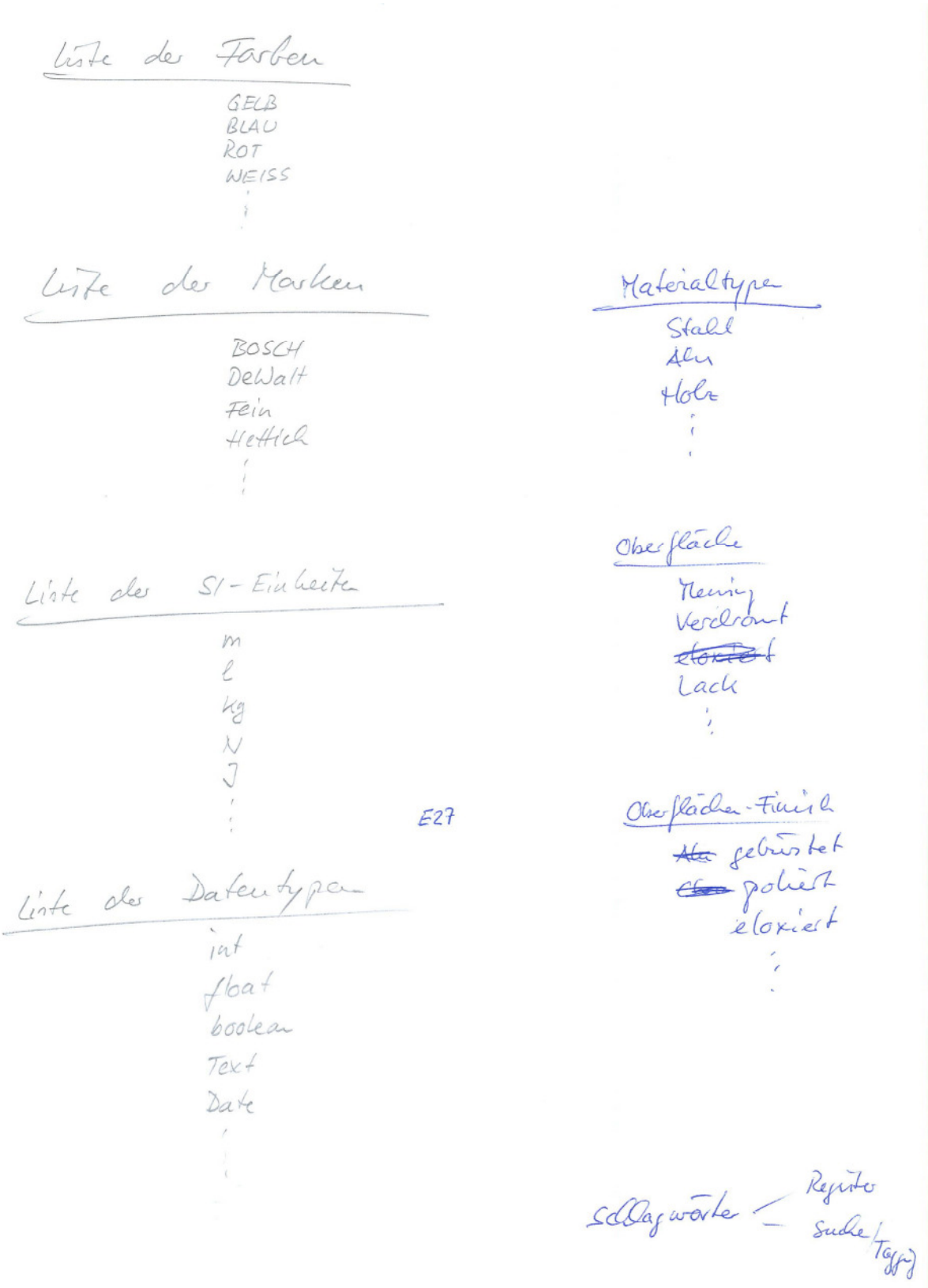
Fragen

Frage	Antwort
Sollen alle Attributre dieselben Metaattribute haben? Soll wir hier flexibel bleiben?	Vereinfachende Annahme: Alle Attribute haben dieselben Attribute.
Wäre es ein wertvolles Feature, wenn die User selbst neue Metaattribute erfassen könnten?	Dies sollte später möglich sein.

Diskussion Architektur

- Design DAL, Schnittstelle DAL
  - Wer ist für Filtern zuständig (formulieren Lambda-Expressions)
  - Objekte aus Kontext lösen (DTO)
    - Navigation Properties
    - wer ist für commit zuständig
    - Einfluss auf Abfragefunktionen

Neuer Input

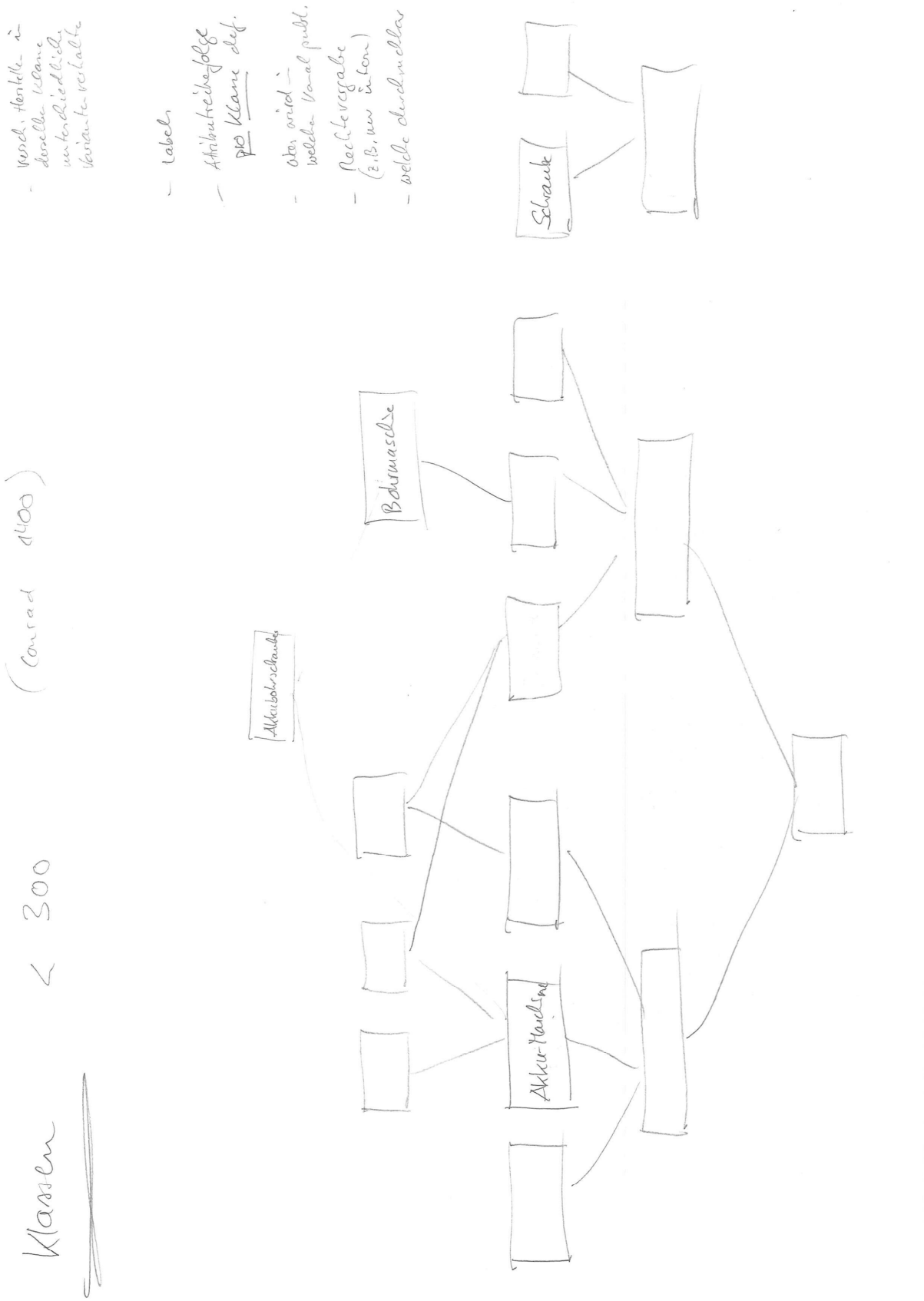

Liste der Farben
GELB
BLAU
ROT
WEISS
Liste der Marken
BOSCH
DeWalt
Fein
Hettich
Liste der SI-Einheiten
m
l
kg
N
J
Liste der Datentypen
int
float
boolean
Text
Date
E27
Materialtype
Stahl
Alu
Holz
Oberfläche
Matt
Verchromt
Lack
Oberflächen-Finish
gebürstet
poliert
eloxiert
Schlagwörter
Register
Suche/Tagging

Attribute > 500

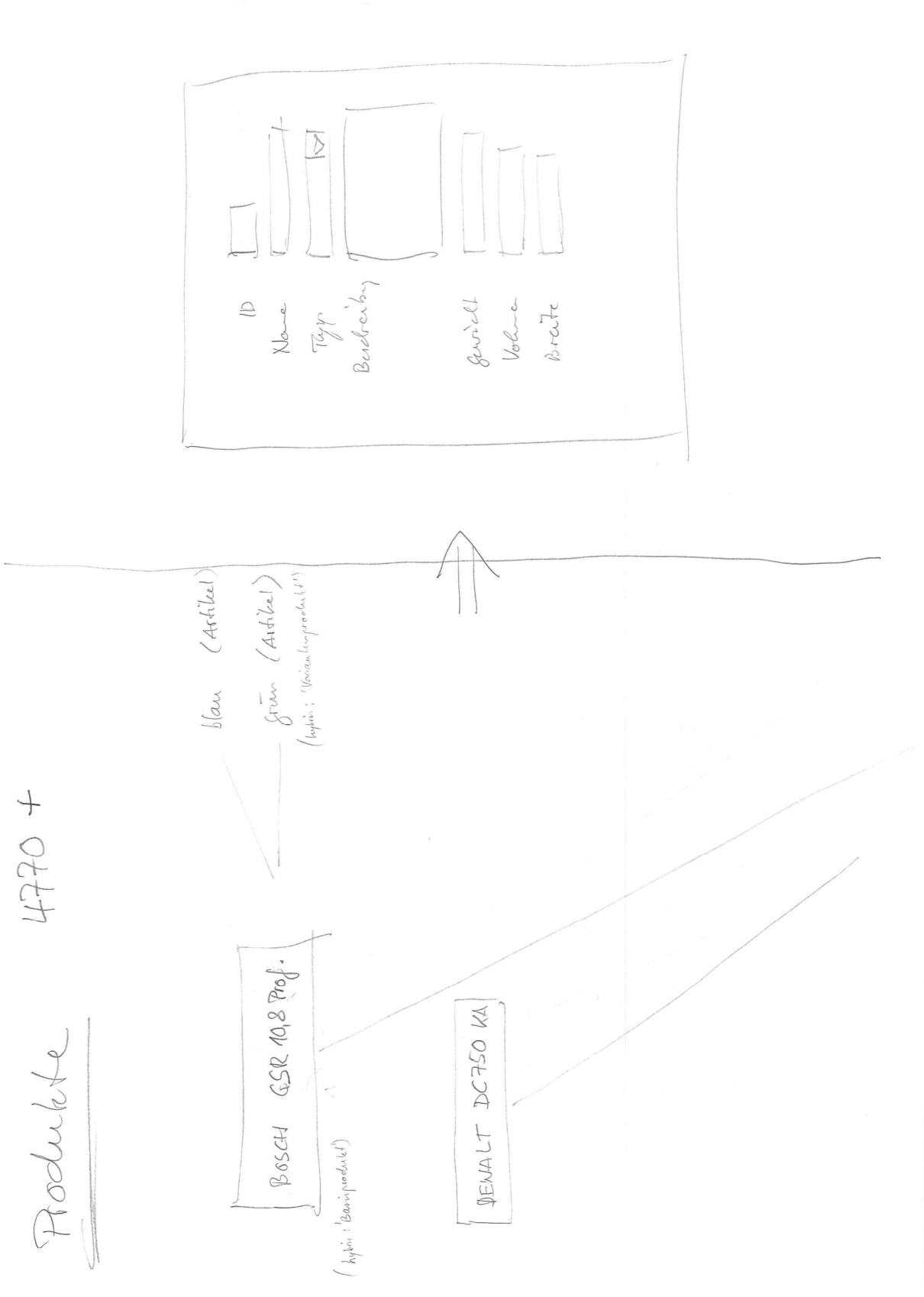
Name	Datentyp	SI Einheit/e	Wertebereich	Muss übersetzt werden
Gewicht	float	g, kg		
Belastung	float	kg, t		
Längenmaß				
Durchmesser				
Distanz				
Oberflächengüte	Aufzählung			
Farbe_Std	Aufzählung			
Marketing-Farbe	Text			
Volumen	float	l, ml		
Einbautiefe				
Stellhöhe				
Drehzahl				
Anzahl Gänge (?)				

Medien

Idee: * Attribute → ist es ein Diskriminator für Artikel oder Produkte



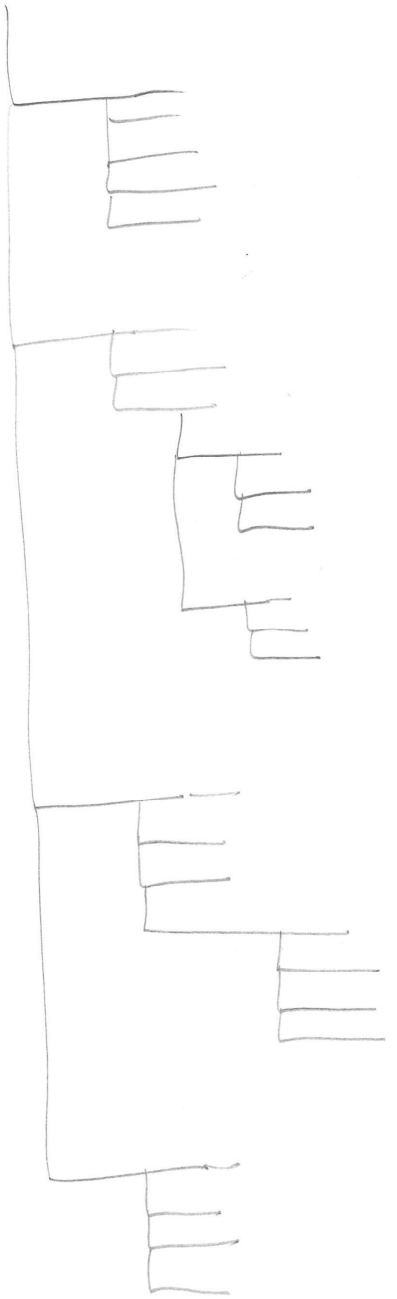
In einer Klasse muss man die Reihenfolge von Attributen (auch die der geerbten Attributen) für die spätere Maske definieren können.

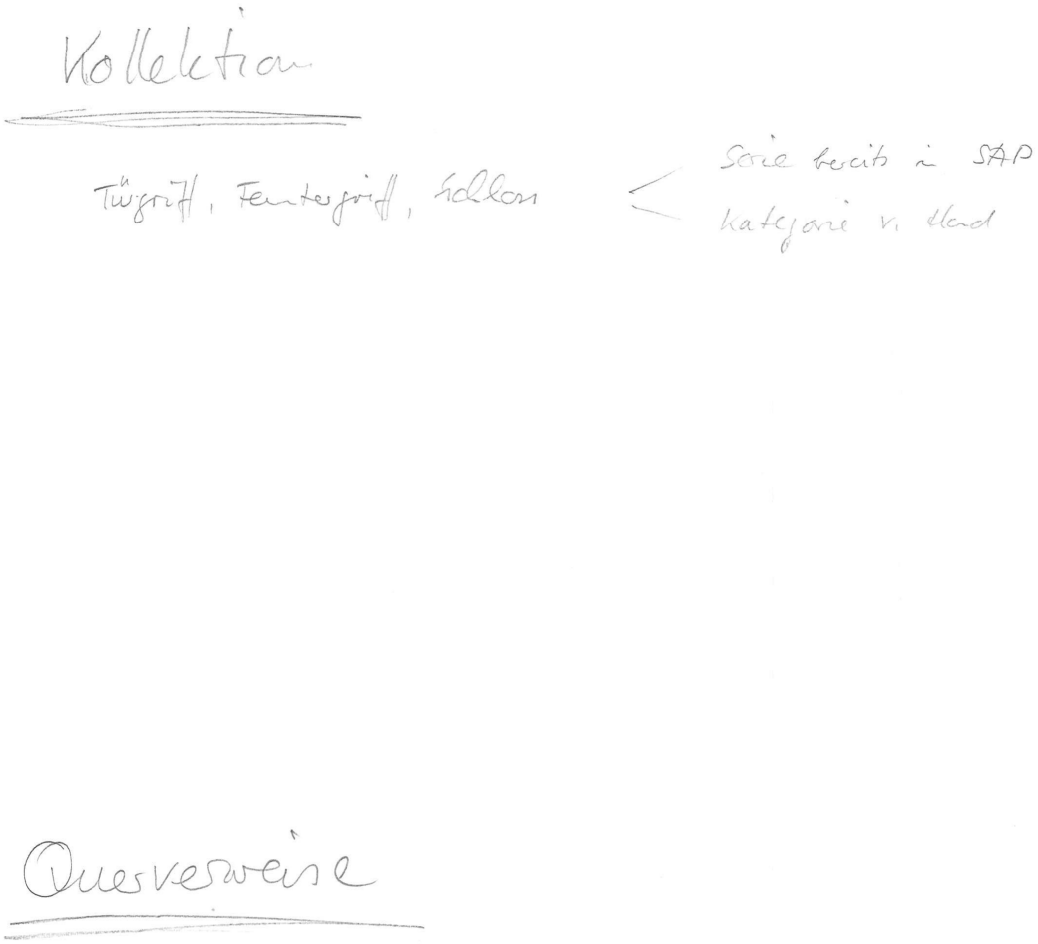


Es gibt ca. 35‘000 Artikel.

Navigationshierarchien

Website, Kataloge




Kollektion
Türgriff, Fenstergriff, Schloss
Serie bereits in SAP
Kategorie vi. Hand
Querverweise

Teamsitzung Samstag, 31. März 2012

Allgemeines

Projektwoche	6
Anwesende	cr, ms
Dauer	20 Min

Rollen

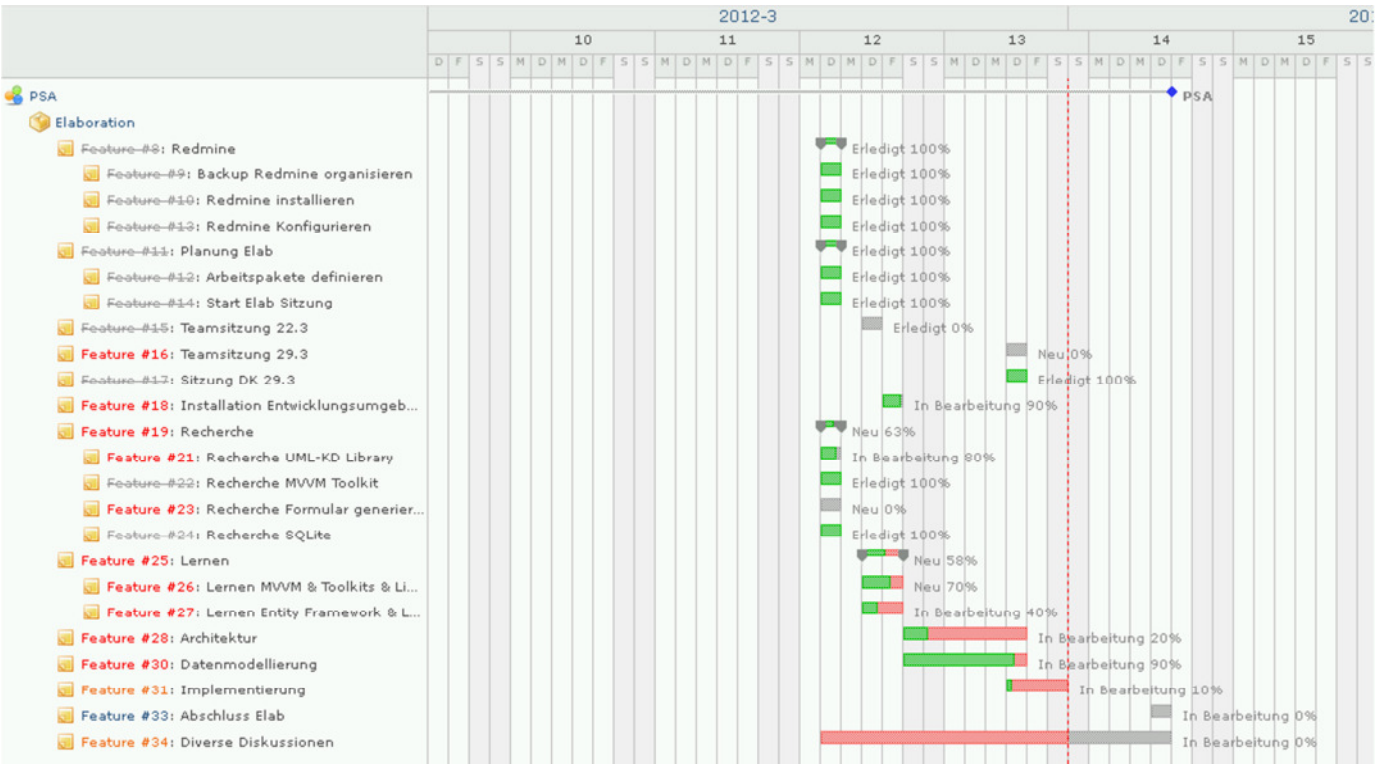
Projekt Manager

Zeitübersicht Gesamtprojekt

Bis Ende Woche sechs hätten wir linear gesehen 270 Stunden arbeiten sollen. Benötigt haben wir bis jetzt 188 Stunden. Somit sind wir 5.125 Arbeitstage / Person im Verzug. Dies entspricht ungefähr unserer Planung.

Zeitplanung Elaboration

Unsere Planung war zu optimistisch: Viele Arbeitspakete benötigten mehr Zeit als geplant.
Obwohl wir Reservezeit eingeplant haben, ist es zurzeit noch unklar, ob wir alle geplanten Features in der Elaboration umsetzen können.



Christoph Rosenberger
Michael Steiner

Team Reflexion

Je nachdem was für eine Art von Arbeit ansteht, ist es uns möglich den Tag besser zu nutzen in dem wir uns den Arbeitsweg sparen. Mit Hilfe von Teamviewer können wir so die meisten Diskussionen abhandeln.

Problem bei Telefonkonferenz: Man sieht den Gesprächspartner nicht und hat so keine Chance seine Reaktionen und Aktionen zu verstehen. Man muss seine Gedankengänge ständig mündlich überliefern.

Iterations-Ende Freitag, 6. April 2012

Allgemeines

Iteration	Elaboration
Dauer (ist)	186 Stunden
Dauer(soll)	177.5 Stunden (+Reserve)

Erreichte Ziele

- Lauffähiges Programm welches durch alle Schichten geht.
- Modellieren der PSA Datenstruktur.
- Dynamische Speicherung von Metaattributen und deren Strukturdefinitionen.
- Anzeige von Attributen und deren Metaattributen.
- Die Infrastruktur ist aufgesetzt und läuft.

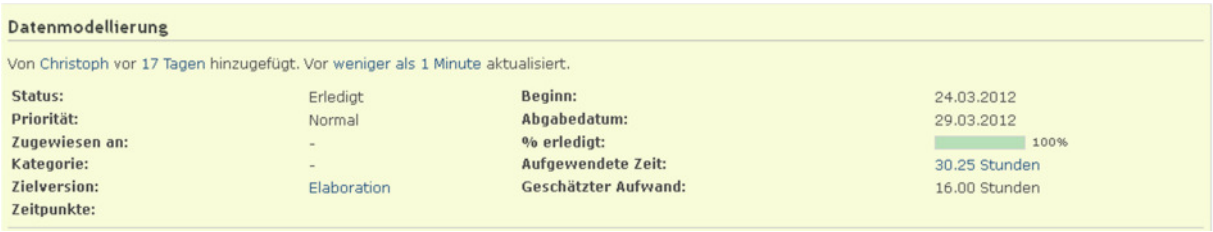
Nicht erreichte Ziele

- Noch nicht alle Metaattribut Typen werden unterstützt (z.B. Rechte)
- Anzeige von Metaattributen in der Attributen Liste.
- Modellierung von Vererbungshierarchien. Dies beinhaltet die Visualisierung und Modellierungsmöglichkeiten über ein Canvas.
- Modellierung von Produkttypen.

Reflexion

Zeitplanung

Dass unsere Zeitplanung sehr optimistisch war, wussten wir von Anfang an. Wir hätten aber schon gehofft, wenigstens die Attributen Ansicht vollständig fertiggestellt zu haben.



Datenmodellierung

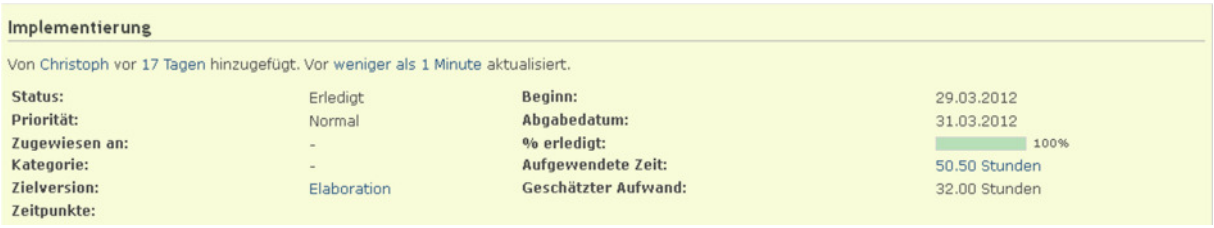
Von Christoph vor 17 Tagen hinzugefügt. Vor weniger als 1 Minute aktualisiert.

Status:	Erledigt	**Beginn:**	24.03.2012
Priorität:	Normal	**Abgabedatum:**	29.03.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	30.25 Stunden
Zielversion:	Elaboration	**Geschätzter Aufwand:**	16.00 Stunden
Zeitpunkte:			

Uns war klar, dass wir kein einfaches Datenmodell vor uns hatten. Wir versuchten eine gute und erweiterbare Lösung zu finden. Diese wurde dann aber zu generisch. Somit versuchten wir es erneut mit einer eher konkreten Lösung. Als wir jedoch merkten, dass man dann gewisse Dinge später nicht

wie gewünscht erfassen kann (z.B. Modellierung von Metaattributen durch den User), wurde aus unserem konkreten Modell ein immer generischeres, bis wir schlussendlich wieder bei unserem allgemein bekannten generischen Modell landeten. So drehten wir uns ein paar Mal im Kreis.

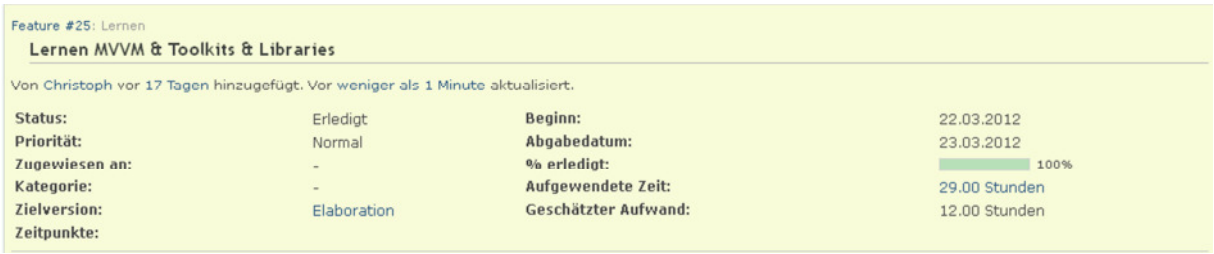
Mit dem aktuellen Resultat sind wir zufrieden und denken, dass diese Zeit doch gut in die Zukunft investiert wurde.



Implementierung

Von Christoph vor 17 Tagen hinzugefügt. Vor weniger als 1 Minute aktualisiert.

Status:	Erledigt	**Beginn:**	29.03.2012
Priorität:	Normal	**Abgabedatum:**	31.03.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	50.50 Stunden
Zielversion:	Elaboration	**Geschätzter Aufwand:**	32.00 Stunden
Zeitpunkte:			



Feature #25: Lernen

Lernen MVVM & Toolkits & Libraries

Von Christoph vor 17 Tagen hinzugefügt. Vor weniger als 1 Minute aktualisiert.

Status:	Erledigt	**Beginn:**	22.03.2012
Priorität:	Normal	**Abgabedatum:**	23.03.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	29.00 Stunden
Zielversion:	Elaboration	**Geschätzter Aufwand:**	12.00 Stunden
Zeitpunkte:			

Wir hatten gehofft, dass wir durch den Besuch von MS-Tech genügend Vorwissen mitbringen. Da wir aber eher spezielle Anforderungen hatten, hatten wir doch mit einigen neuen Problemen zu kämpfen. Dies hat sich auf die Implementierung sowie auf die Recherche stark ausgewirkt.

Teamarbeit

Pair-programming

Da wir laufend grundlegende Entscheide treffen mussten und wir beide das erste Mal mit diesen neuen Technologien arbeiteten, entschlossen wir uns zu einem Pairprogramming-Ansatz.

(Driver: Tastaturbediener, Navigator: Beobachter)

- Der Navigator soll einen guten Zeitpunkt finden um dem Driver seinen Input zu geben, so dass er den Driver optimal unterstützt und nicht unterbricht oder stört.
- Es ist wichtig, dass der Navigator die aktuellen Tätigkeiten des Drivers versteht, so dass er ihn (in einem günstigen Zeitpunkt) auf Fehler oder Optimierungen hinweisen kann.
- Der Driver muss den Navigator ständig informieren, welches mini-Arbeitspaket er nun implementieren möchte und welche Schritte dies beinhaltet. Dadurch ist es für den Navigator einfacher zu Folgen und hat somit früher die Möglichkeit einen besseren und korrigierenden Input zu geben.
- Für den Navigator ist es wichtig, dass er ein gutes Notizwerkzeug hat, so dass er bei einem Fehler den Driver nicht unterbrechen muss, sondern ihm später (in einem günstigen Zeitpunkt), die Fehler und Verbesserungsvorschläge mitteilen kann.

Fern Team Arbeit

Zurzeit arbeiten wir per Telefon und Teamviewer. Dies erspart uns den

Arbeitsweg (ca. 2.5 Stunden), wodurch mehr Zeit für die eigentliche Arbeit bleibt und den Lärmpegel, welcher in der Schule zum Teil stören kann.

Punkte an denen wir noch Arbeiten können

- Den Anderen ständig auf dem Laufenden halten, wenn man einer neuen Tätigkeit (wie z.B. etwas googlen) nachgeht.
- Vermehrte kurze (akustische!) Feedbacks, dass man etwas verstanden hat (da wir keine Visuelle Kommunikation haben)

Sitzung Donnerstag, 5. April 2012

Allgemeines

Projektwoche 7

Anwesende cr, ms, dk

Dauer 2

Traktanden

- Datenmodell Review

Neuer Input

- Vererbung könnte man in den Masken z.B. mit Farben anzeigen
- Die Maske sollte Platzökonomisch implementiert werden
    - Die Maske muss nicht unbedingt speziell schön sein
    - Die Maske sollte dem Kunden zeigen, was für Daten er später erfassen muss
- Es sollten Beispieldaten für die Masken erfasst werden können
- Es gibt eine (Mehrfach)Vererbungshierarchie mit ca. 5 Stufen
- Unser Programm wird von Herrn Keller auf ca. 4000 Codezeilen geschätzt
- Einer der Bildschirme des Kunden hat eine Auflösung von 1280x1024

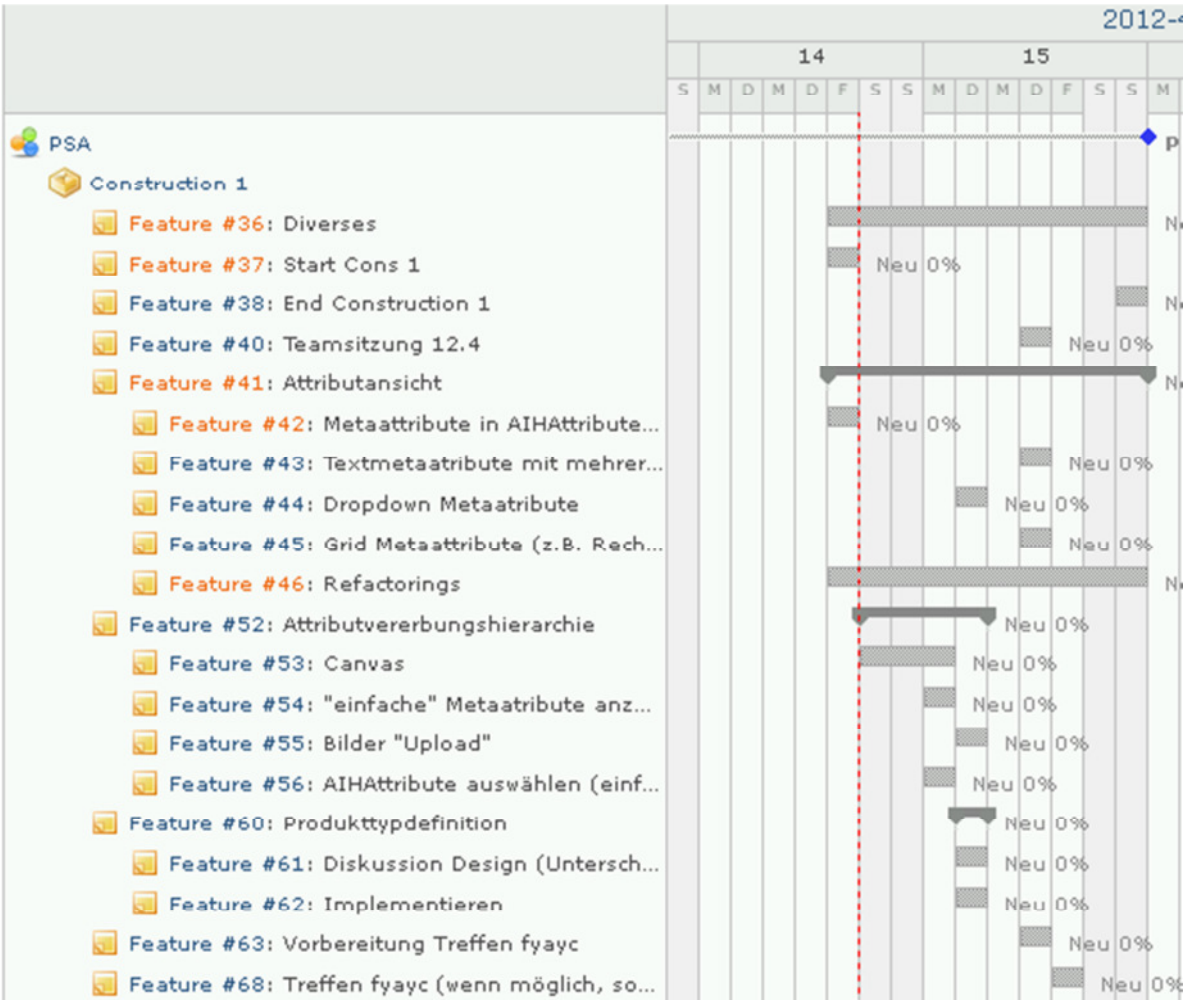
Start Construction 1, 06.04.2012

Allgemeines

Iteration	Construction 1
Projektwochen	7 bis 8
Datum	06.04.2012 bis 15.04.2012
Dauer (soll)	103 Stunden

Iterationsziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand
36		Normal	Diverses		06.04.2012	15.04.2012	8.0
37		Normal	Start Cons 1		06.04.2012	06.04.2012	5.0
38		Normal	End Construction 1		15.04.2012	15.04.2012	10.0
40		Niedrig	Teamsitzung 12.4		12.04.2012	12.04.2012	2.0
41		Dringend	Attributansicht		06.04.2012	15.04.2012	30.0
42	Feature #41	Hoch	▸ Metaattribute in AIHAttributeList anzeigen		06.04.2012	06.04.2012	4.0
43	Feature #41	Niedrig	▸ Textmetaatribute mit mehreren Zeilen		12.04.2012	12.04.2012	2.0
44	Feature #41	Niedrig	▸ Dropdown Metaatribute		10.04.2012	10.04.2012	6.0
45	Feature #41	Niedrig	▸ Grid Metaattribute (z.B. Rechte)		12.04.2012	12.04.2012	10.0
46	Feature #41	Dringend	▸ Refactorings		06.04.2012	15.04.2012	8.0
52		Dringend	Attributvererbungshierarchie		07.04.2012	10.04.2012	30.0
53	Feature #52	Hoch	▸ Canvas		07.04.2012	09.04.2012	20.0
54	Feature #52	Hoch	▸ "einfache" Metaatribute anzeigen		09.04.2012	09.04.2012	2.0
55	Feature #52	Normal	▸ Bilder "Upload"		10.04.2012	10.04.2012	3.0
56	Feature #52	Dringend	▸ AIHAttribute auswählen (einfache Version)		09.04.2012	09.04.2012	5.0
60		Normal	Produkttypdefinition		10.04.2012	10.04.2012	8.0
61	Feature #60	Normal	▸ Diskussion Design (Unterschied Attributvererbungshierarchie, Vererbung innerhalb Produkttypen)		10.04.2012	10.04.2012	2.0
62	Feature #60	Normal	▸ Implementieren		10.04.2012	10.04.2012	6.0
63		Hoch	Vorbereitung Treffen fyayc		12.04.2012	12.04.2012	4.0
68		Normal	Treffen fyayc (wenn möglich, sonst Treffen DK)		13.04.2012	13.04.2012	6.0



Rollen

Projekt Manager

Treffen fyayc

Ende Projektwoche 8 ist die Hälfte des Semesters vorbei. Wenn wir unsere Planung einhalten, haben wir ein erstes Release, welches wir mit gutem Gewissen bei fyayc präsentieren können.

Die Planung ist eng gepackt und sicherlich sehr Ambitiös. Auf der anderen Seite haben wir Arbeitspakete mit niedriger Priorität im Umfang von 20 Stunden. Wir denken also, dass wir bis am Freitag dem 13.04.2012 eine Zeigbare Version haben werden.

Bei entsprechendem Erfolg in der Umsetzung werden wir am Dienstag dem 10.04.2012 versuchen, ein Treffen bei fyayc auf den 13.04.2012 zu organisieren.

Risiko Manager

Canvas

Beschreibung

Wir haben beide keine Erfahrung, wie man mit einer Canvas umgeht. Auch wissen wir noch nicht genau, wie man im MVVM Pattern eine Canvas "ansteuert".

Wir haben bereits Recherchen in der Elaboration gemacht. Leider haben wir keine Library gefunden, welche uns die UML Modellierung abnehmen oder erleichtern würde.

Mögliche Folgen

- Zusätzlicher lern Aufwand um mit der Umsetzung zu beginnen.
- Grössere Probleme bei der Umsetzung
- Mehr als die geplanten 20 Stunden sind nötig

Mögliche Lösung

Die Bedienung der Canvas auf imperativ zu lösen, scheint nicht besonders schwer zu sein; dies würde aber dem MVVM Ansatz wiedersprechen. Ein Lösungsansatz wäre, oberhalb eines ViewModel eine Klasse einzubauen, welche die Canvas kennt und direkt ansteuern kann.

Sitzung 12.04.2012

Allgemeines

Projektwoche	8
Anwesende	cr, ms, dk
Dauer	1.5h

Traktanden

- Treffen fyayc am besten in einer Woche. Dann werden wir unser Programm mit WireFrames "vollständig" designt haben / alle Bereiche werden sichtbar sein.
- Design by Contract ok? Oder Zeitverschwendung? (CodeContracts)
- Benötigen wir ein Login? (Damit man nachverfolgen kann wer welche Änderungen getätigt hat)
- Soll das Programm auf den Kunden weitergegeben werden können? Wenn ja, benötigt es ein Lizenzsystem?
- Wo liegt der Wert des Programms? Viele Features? User freundliches GUI?
- Wo sollen wir die Bilder speichern?
- Produkttyp anstatt eigene Ansicht, ein Flag, welches eine Abstrakte-Klasse als Produkttyp deklariert?

Neuer Input

- Treffen mit fyayc in einer Woche (Donnerstag oder Freitag)
- Design by Contract für DAL ev. sinnvoll
- Login wird später ev. benötigt (jetzt vernachlässigen)
- Ja, das Programm wird ev. dem Kunden weitergegeben, jedoch im Moment ohne Lizenz
- Sauberer Code ist wichtig, viele Features sind wichtig, Programm soll bedienbar sein → Muss nicht übertrieben gut sein, lieber mehr Features
- Bilder relativ im Programmverzeichnis ablegen
- Produkttyp geht so in Ordnung
- Es wird ein „Aktuellen Stand“ speichern-Button benötigt, welcher ein DB-Dump erstellt
- Beim fyayc Treffen sollte ein xml/csv Export möglich sein
- Beim fyayc Treffen sollte die Maskengenerierung gezeigt werden können

Neue Termine

- 19/20. April Treffen fyayc

Christoph Rosenberger
Michael Steiner

Iterations-Ende 14.04.2012

Allgemeines

Iteration Construction 1

Dauer (ist) 96.5 Stunden (Iteration wurde abgekürzt)

Dauer (soll) 103 Stunden

Neu Planung

Treffen fyayc In dieser Iteration hätten wir bereits den Projekt Vorschritt bei fyayc präsentieren können; wir haben ein erstes laufendes Release. Um aus diesem Treffen möglichst viel nützliches Feedback zu bekommen, haben wir beschlossen, dieses um eine Woche zu verschieben.

Dafür werden wir in der nächsten Iteration die Komplette Navigationsstruktur im Programm umsetzen und die noch nicht programmierten Features mit Wireframes darstellen. Dadurch können wir einen Überblick über das gesamte Programm geben. Wir möchten damit den potentiellen Usern zeigen was geplant ist und welche Möglichkeiten wir ihnen bieten wollen. Dadurch soll auch verhindert werden, dass in der Diskussion Features gewünscht werden, welche bereits geplant sind.

Arbeitsfortschritt Den Code haben wir soweit entwickelt, dass wir ohne Probleme in einer späteren Iteration die Features fertigstellen können. Einige Ideen sind erst teilweise umgesetzt, daher haben wir auf die Dokumentation verzichtet. Wir werden in Construction 3 diese Arbeiten beenden.

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
36		Normal	Diverses		06.04.2012	15.04.2012	8.0	
37		Normal	Start Cons 1		06.04.2012	06.04.2012	5.0	
38		Normal	End Construction 1		15.04.2012	15.04.2012	10.0	
40		Niedrig	Teamsitzung 12.4		12.04.2012	12.04.2012	2.0	
41		Dringend	Attributansicht		06.04.2012	15.04.2012	30.0	
42	~~Feature #41~~	Hoch	Metaattribute in AIHAttributeList anzeigen		06.04.2012	06.04.2012	4.0	
43	~~Feature #41~~	Niedrig	Textmetaatribute mit mehreren Zeilen		12.04.2012	12.04.2012	2.0	
44	~~Feature #41~~	Niedrig	Dropdown Metaatribute		10.04.2012	10.04.2012	6.0	
45	~~Feature #41~~	Niedrig	Grid Metaattribute (z.B. Rechte)		12.04.2012	12.04.2012	10.0	
46	~~Feature #41~~	Dringend	Refactorings		06.04.2012	15.04.2012	8.0	
52		Dringend	Attributvererbungshierarchie		07.04.2012	10.04.2012	30.0	
53	~~Feature #52~~	Hoch	Canvas		07.04.2012	09.04.2012	20.0	
54	~~Feature #52~~	Hoch	"einfache" Metaatribute anzeigen		09.04.2012	09.04.2012	2.0	
55	~~Feature #52~~	Normal	Bilder "Upload"		10.04.2012	10.04.2012	3.0	
56	~~Feature #52~~	Dringend	AIHAttribute auswählen (einfache Version)		09.04.2012	09.04.2012	5.0	
60		Normal	Produkttypdefinition		10.04.2012	10.04.2012	8.0	
61	~~Feature #60~~	Normal	Diskussion Design (Unterschied Attributvererbungshierarchie, Vererbung innerhalb Produkttypen)		10.04.2012	10.04.2012	2.0	
62	~~Feature #60~~	Normal	Implementieren		10.04.2012	10.04.2012	6.0	
63		Hoch	Vorbereitung Treffen fyayc		12.04.2012	12.04.2012	4.0	
68		Normal	Treffen fyayc (wenn möglich, sonst Treffen DK)		13.04.2012	13.04.2012	6.0	

Reflexion

WPF Wir hatten einige Probleme mit WPF. Während einfache Beispiele mit WPF und MVVM sehr einfach umsetzbar sind, mussten wir für ein wenig mehr Komplexität verhältnismässig viel Zeit investieren. Hier zeigt sich unsere kaum vorhandene Erfahrung mit dieser

Technologie.

Dropdown Metaatribute

Von Christoph vor 8 Tagen hinzugefügt. Vor etwa 1 Stunde aktualisiert.

Status:	Erledigt	**Beginn:**	10.04.2012
Priorität:	Niedrig	**Abgabedatum:**	10.04.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	10.75 Stunden
Zielversion:	Construction 1	**Geschätzter Aufwand:**	6.00 Stunden
Zeitpunkte:			

Datenmodell

Das Datenmodell haben wir bereits in der Elaboration erstellt. In dieser Iteration haben wir nun erkannt, wie man mit diesem Datenmodell umgehen kann. Dabei haben wir auch nochmals eine Alternative durchbesprochen. Es zeigte sich aber, dass unsere Variante gut ist.

Metaattribute in AlHAttributeList anzeigen

Von Christoph vor 8 Tagen hinzugefügt. Vor etwa 1 Stunde aktualisiert.

Status:	Erledigt	**Beginn:**	06.04.2012
Priorität:	Hoch	**Abgabedatum:**	06.04.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	11.75 Stunden
Zielversion:	Construction 1	**Geschätzter Aufwand:**	4.00 Stunden
Zeitpunkte:			

UML Editor

Auch bei der Zeitschätzung für den UML Editor waren wir zu optimistisch.

Canvas

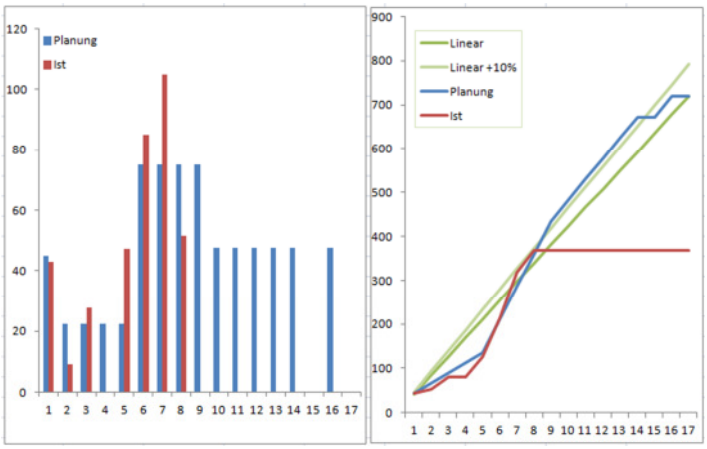
Von Christoph vor 8 Tagen hinzugefügt. Vor etwa 1 Stunde aktualisiert.

Status:	Erledigt	**Beginn:**	07.04.2012
Priorität:	Hoch	**Abgabedatum:**	09.04.2012
Zugewiesen an:	-	**% erledigt:**	80%
Kategorie:	-	**Aufgewendete Zeit:**	34.75 Stunden
Zielversion:	Construction 1	**Geschätzter Aufwand:**	20.00 Stunden
Zeitpunkte:			

Rollen

Projekt Manager

Um die Iterationsziele zu erreichen, haben wir viel gearbeitet. Dies zeigt sich auch in der Gesamt-Zeitauswertung. (Zeiten der Woche 8 bis Donnerstag erfasst.)



Iterations-Start 14.04.2012

Allgemeines

Iteration	Construction 2
Projektwochen	8 bis 9
Datum	12.04.2012 bis 20.04.2012
Dauer (soll)	78 Stunden

Iterationsziele

Inhalt

Ziel dieser Iteration ist die Vorbereitung und Durchführung einer Präsentation bei fyayc. Wir möchten die gesamte geplante Funktionalität von PSA präsentieren können. Dazu erstellen wir die komplette Navigation. Diejenigen Features welche noch nicht implementiert sind, werden wir mit Wireframes darstellen.

Nutzen

Bei diesem Treffen geht es sowohl darum, fyayc den Vorschritt des Projekts zu zeigen, als auch ein Feedback zu erhalten. Dadurch können wir sicherstellen, dass wir die richtigen Features richtig geplant haben.

Wir erhoffen uns allfälliger weise korrigierende und erweiternde Kritiken. Ausserdem werden wir anhand der Diskussion in folgenden Iterationen die Prioritäten besser setzen können.

Detaildiskussionen einzelner Funktionen werden wir aber in dieser Sitzung kaum klären können; für diese werden wir weiterhin auf den guten Input von Herrn Keller angewiesen sein.

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
69		Sofort	Analyse		13.04.2012	15.04.2012	12.0	
70	Feature #69	Sofort	Screens		13.04.2012	15.04.2012	7.0	
71	Feature #69	Sofort	Screeninhalt		13.04.2012	15.04.2012	5.0	
72		Dringend	Designen		13.04.2012	15.04.2012	11.0	
73	Feature #72	Dringend	Navigation, Grundprinzipien		13.04.2012	15.04.2012	2.0	
74	Feature #72	Dringend	Wireframes zeichnen		13.04.2012	15.04.2012	9.0	
75		Dringend	Implementieren		13.04.2012	14.04.2012	6.0	
76	Feature #75	Dringend	Neue GUI Struktur umsetzen		13.04.2012	14.04.2012	2.0	
77	Feature #75	Dringend	Bestehenden Code in neue Struktur einbinden		13.04.2012	14.04.2012	2.0	
78	Feature #75	Dringend	Wireframes einbinden		13.04.2012	14.04.2012	2.0	
79		Hoch	Beispiele		13.04.2012	17.04.2012	8.0	
80	Feature #79	Hoch	Beispiele von fyayc ergänzen und anpassen auf unsere Darstellungen		13.04.2012	17.04.2012	4.0	
81	Feature #79	Hoch	Beispiele in PSA erfassen		16.04.2012	17.04.2012	2.0	
82	Feature #79	Hoch	Beispiele in Wireframes einzeichnen		16.04.2012	17.04.2012	2.0	
83		Normal	Präsentation vorbereiten		17.04.2012	18.04.2012	21.0	
84	Feature #83	Normal	Durchspielen		17.04.2012	17.04.2012	4.0	
85	Feature #83	Normal	Sitemap erstellen		17.04.2012	17.04.2012	1.0	
86	Feature #83	Niedrig	Eigene Themen einbringen		17.04.2012	18.04.2012	16.0	
87		Hoch	Dokumentation		13.04.2012	20.04.2012	12.0	
88	Feature #87	Normal	Planung dokumentieren		13.04.2012	14.04.2012	2.0	
89	Feature #87	Hoch	Sitzung dokumentieren		20.04.2012	20.04.2012	4.0	
90	Feature #87	Normal	Abschluss dokumentieren (Abschlusssitzung)		20.04.2012	20.04.2012	2.0	
91	Feature #87	Normal	Eigentliche Arbeit dokumentieren		20.04.2012	20.04.2012	4.0	
92		Niedrig	Programm verschönern		18.04.2012	18.04.2012		
93	Feature #92	Niedrig	Features fertigstellen		18.04.2012	18.04.2012		
94		Normal	Präsentation bei fyayc		19.04.2012	19.04.2012	6.0	

Abbildung 6: Arbeitspakete Construction 2

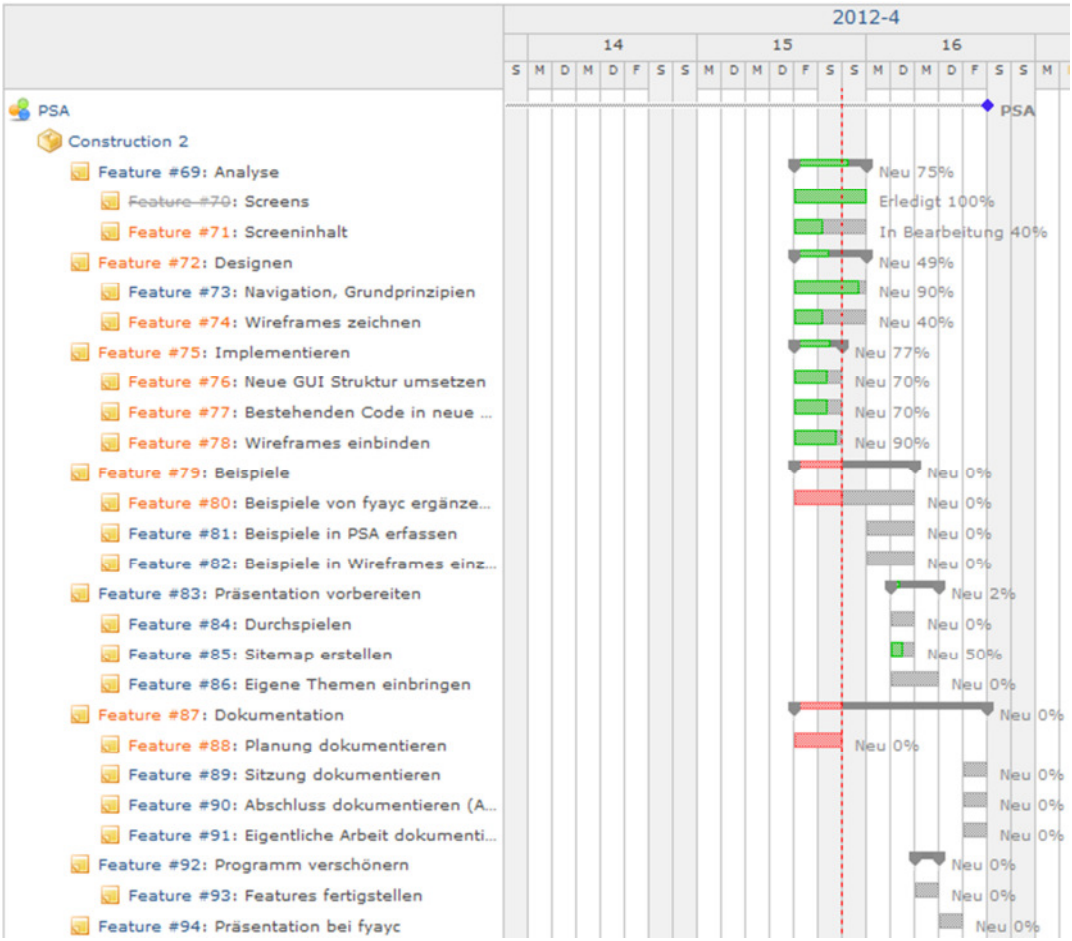


Abbildung 7: Gant Chart Construction 2

Priorisierung der Arbeitspakete

Für die Arbeitspakete welche vor dem Treffen bei fyayc umgesetzt werden müssten, haben wir eigentlich zu wenig Zeit. Für diejenigen Arbeitspakete mit niedriger Priorität werden wir leider nicht die gewünschte Zeit investieren

können; insbesondere werden wir wohl kaum vertieft auf unsere eigenen Ideen eingehen können.

Sitzung 17.04.2012

Allgemeines

Projektwoche 9

Anwesende cr, ms, dk

Dauer 45 min

Fragen

Frage	Resultat
Darstellung der vererbten Attribute wenn: • wenn ein Attribut (dank mehrfach Vererbung) mehrfach geerbt wird? • wenn ein Attribut einer Klasse selbst, aber auch einer Vaterklasse zugewiesen wird?	Auf keinen Fall Attribute mehrfach anzeigen. Fall 2: Attribut aus Sohnklasse löschen (u.U. mit Warnmeldung)
Viele Metaattribute -> Viel Aufwand. Ist unsere Auswahl an Metaattributen schon zuviel?	Nein, ist ok.
Tab Kollektions und Querverweise	Um diese sinnvoll zu füllen, bräuchte man produktiv Daten. Dies kann also nicht in PSA geschehen.

Eigene Themen einbringen ?
• Eigene Tabs
  ○ Frontend Visualisierung
  ○ Mapping Channels
  ○ Mapping Umsysteme
  ○ Statistiken / Kostenanalyse
  ○ Facetted Search
• Korrelierende Attribute
• Weiterführende Diskussion über Datentypmodellierung
  ○ (Ob ein singulärer Wert (100 Umdrehungen/Minute) oder ein Range (50 - 150 Umdr. /Min) oder mehrere Werte (50, 100, 150 Umdr. /Min) ist zwingend Abhängig von AIHAttribut, sondern von konkretem Produkt bzw. dessen Variationen. Oder sollen wir Textfeld einfach lassen?
  ○ Alles soll in SI-Einheiten gespeichert werden. Dafür würden wir ein Feld Anzeigedatentyp anbieten (z.B. Datentyp: Meter, Anzeigedatentyp: Millimeter). Dazu braucht es dann natürlich die entsprechende Konvertierungslogik.
• (Semantischer Typ eher nicht, wir wissen selbst viel zu wenig, auf was das hinauslaufen würde... (Bsp. Innenbreite, Aussenbreite, Seitenbreite -> alles Breiten))
	Eigene Tabs: sehr gut.
	
	Weitere Themen: Wenn es sich in der Diskussion anbietet. Sonst eher nicht. Eher einfach zuhören, was Jonathan von sich aus erzählt.

Diskussion über Prioritäten? Oder sollen wir die jeweils mit DK besprechen (da hierzu auch Zeitschätzungen nötig wären)	Siehe oben.

Christoph Rosenberger
Michael Steiner

Sitzung bei fyayc 19.04.2012

Allgemeines

Projektwoche 9

Anwesende Jonathan Möller, cr, ms, dk,

Dauer 1 Stunde

Gesprächslog

(Falls keine Person angegeben, kam der Input von JM)

- 8:05: Kann ein Attribut bei mehreren Klassen vorkommen: Das kann man so machen. Die Frage ist, versteht das eine Sekretärin?
- 11:20: "Variation" muss pro Attribut-Produkttyp Kombination erfassbar sein. / Beim Produkttyp muss man sagen, welche Attribute eine Variante ausmachen.
    - 12:40: Es kann mehrere Dimensionen bei den Varianten Geben (z.B. Farbe und Grösse)
    - Die Dimensionen können von Hersteller zu Hersteller unterschiedlich sein.
    - 13:00: Entscheidung was eine Variante ist, ist vollkommen willkürlich. Die Frage ist, wie glaubt man, dass ein Produkt im Shop dargestellt wird.
    - 14:15, 15:35: Variationsdiskussion hat extrem hohen Wert.
    - 15:45: Die Diskussion, was eine Variante ist, findet auf den Produkttypen statt.
    - 16:15: Die Leute sollen erleben, was die unterschiedlichen Variantendimensionen sind
    - 16:30: Das Maximum an Dimensionen hat JM bei Pneus erlebt: Es waren 8 Dimensionen
- 15:10: Gewisse Marken sind so wichtig, dass man extra eigene Produkttypen erstellt.
- 17:10: Reihenfolge der Attribute muss pro Produkttyp festgelegt werden können.
    - Man soll den Produkttyp möglichst so designen können, wie man ihn später auch erlebt.
- 19:10: JM findet die abstrakten Produkttypen gut. Sie gruppieren die Attribute.
    - 19:25, DK: Die Frage ist, ob dies noch übersichtlich ist mit 300 Klassen
    - JM: Es bräuchte Filter
    - DK: Man muss scrollen können
    - CR: Es braucht Packages
- 20:40: Zuerst Attribute sammel n und Produkttypen sammeln. Dann den Produkttypen Attribute zuweisen.
    - JM hatte mal einen Mitarbeiter, der hat behauptet, dass man den Klassenbaum automatisch generieren kann.
    - Es gibt schon Leute, die wollen verstehen, wie diese Struktur ist. Dies ist die Minderheit.

    - 22:15: Das System soll erkennen, welches gleiche oder ähnliche Typen sind. Der User soll dann entscheiden, wie er dies handeln möchte.
    - 24:00, 24:40: Man soll auch bei den Attributen sagen, zu welchen Produkttypen sie gehören (z.B. gehört zu allen Produkttypen)
    - 23:55: Zuordnen von Attributen zu Produkttypen wird "von einfach gestrickten Leuten" gemacht
    - Zuordnen von Attributen zu Produkttypen soll intuitiv und schnell gehen.
    - Wenn ich ein Attribut zu einem Produkttyp hinzufüge, soll PSA vorschlagen, welche Attribute sonst noch passen würden.
    - Es gibt Attribute, die fallen den Leuten schnell ein. Um ein solches Attribut muss man dann herumfragen, ob es da noch weitere Attribute gibt.
- 16:40: Reihenfolge der Attribute ist sehr wichtig
    - Es sollen zuerst alle wichtigen Attribute befüllt werden, von allen Produkten. Und nicht etwa zuerst ein Produkt fertig stellen.
    - 28:50 Mittelfristig könnte man Spalten machen: Von wo bis wo ist es sehr kurz (Anzeige mit sehr wenigen Attributen), von wo bis wo ist kurz, von wo bis wo ist lang, der Rest ist alles. Damit könnte man ein Template hinterlegen und ein Preview für die verschiedenen Devices.
- 30:40 Diskussion wegen Aufwand um Metaattribute abzufüllen
    - Die Varianz innerhalb des Attributes ist nicht so gross, es gibt vielleicht 4 Unterschiedliche Befüllungen für ein Attribut über die verschiedenen Produkttypen hinweg.
        - Z.B. ist Internationalisierbar bei 90% der Produkte gleich.
    - 32:20 Sehr sexy wäre, wenn man Templates hinterlegen könnte, wie verschiedene Darstellungen (z.B. Suchresultat) aussehen werden.
- 35:45 Es gibt Attribute, die dürfen nur intern sichtbar sein (z.B. Marche, ist es ein Produkt das gepuscht werden soll, Verfügbarkeit, ...)
- 36:40 Quelle (ERP, ...) ist sehr wichtig. Man muss wissen, was automatisiert und was von Hand erfasst werden muss.
- 37:00 Kosten anzeigen, wie viel Zeit braucht es um ein neues Produkt eines Produkttyps zu erfassen? Dazu dann eine Ampel machen: Produkttypen welche zu viel Zeit brauchen, werden rot.
    - Aus Quelle und Attributtyp könnte Hypothese generiert werden.
    - Anzeige als Schweizer Franken wäre super.
- 43:30 Grösster Nutzen
    - Vorschau mit Templates (wie sieht z.B. eine Suchresultat Zeile aus)
    - Zuordnung der Attribute von beiden Seiten (von Attribut aus aber auch von Produkttyp aus). Dies muss schnell gehen.
    - Reihenfolge der Attribute pro Produkttyp
    - eher zweitrangig:
        - Kostenberechnung
        - 45:55: automatisches Attribute gruppieren, wobei dies ein sehr wertvolles Feature werden kann.
        - Vorschlaglogik, welche Attribute noch passen könnten.
        - Identifizieren von ähnlichen Produkttypen.

- 48:20: Von einander abhängige Attribute / Korrelierende Attribute
  - Farbe: Es gibt Marketingfarbe und kommunizierter / vergleichbarer Farbwert. Diese stehen in Relation zu einander.
  - Zusammengesetze Attribute: Z.B. der Name setzt sich zusammen aus Produkttyp, Name und Bezeichnung. Daran lässt sich ein Produkt identifizieren.
  - Marketingtexte: Texte welche Platzhalter für Attribute hat. (Sowas müssen wir nicht abbilden können, das muss das PIM können)
  - Umrechnen von cm zu Inch. Z.B. anhand einer Referenz Tabelle (es sind also nicht alle Werte als Output möglich). Auch Kleidergrössen wären denkbar.

Sitzung 26.04.2012

Allgemeines

Projektwoche 9

Anwesende cr, ms, dk

Dauer 1.75h

Themen aus E-Mail

Frage	Antwort
Was ist der Mehrwert von PSA?	Es gibt noch kein vergleichbares Tool. Der Mehrwert besteht aus einem Funktionierenden und lauffähigem Tool. Die Usability soll brauchbar sein. (Siehe auch Sitzung vom 22.02.2012 Abschnitt „Fragen“: Damals wurde der Mehrwert eher in einem proof of Concept gesehen um gute Ideen aufzuzeigen.)
Wie sieht das Einsatzscenario von PSA aus?	Fyayc geht zum Kunden. Gemeinsam erfassen sie die ersten Attribute und später auch Produkttypen. Das Tool wird auf einem Beamer projiziert und die Daten werden gemeinsam eingetragen. Es geht auch darum, die gewonnenen Erkenntnisse aufzuschreiben und zu zeigen.
Wer sind die User von PSA? Welches Wissen und welche Fähigkeiten können wir von ihnen erwarten?	Das Tool wird von einem Ingenieur bedient.
Gehen wir richtig in der Annahme, dass eine Anbindung an ein PIM nicht möglich ist?	Ja, es kann nicht an ein PIM angebunden werden, da diese heute noch nicht so weit entwickelt sind, als dass sie eine Schnittstelle anbieten würden.
Wie stellen sie sich den Export vor?	Alle Klassen, alle Attribute in beliebiger Form (beliebiges xml, csv).

Inputs von Jonathan

Input	**Antwort DK**
Topdown Ansatz (inkl. Generierung von abstrakten Klassen)	Nein. Schön wäre wenn Refactorings durchgeführt werden könnten und Refactoring-vorschläge gemacht werden würden.
Metaattribute (insbesondere „Unterscheidet Variationen“) pro Produkttyp und Attribut	Die Unterscheidung von Varianten ist zu aufwändig und kann im Moment ignoriert werden.

Neu entstandene Ideen

Siehe „Modellierung via Präsentationsmatrix“.

Priorisierung (welche „alten“ Features können wir streichen)

- Komplexe Metaattribute
- UML-Editor (genügt ein Generator?)
- Suche von Attributen & Klassen
- Modellieren von Navigationshierarchien
- UndoRedo
- PSA übersetzbar machen

Neuer Input von DK

Erzählte von seinem letzten Kundenbesuch: UML alleine wird nicht funktionieren, Mehrfachvererbung wird nur zum Teil benötigt. Es braucht eine Ansicht wo man viele Klassen sieht, dafür die Attribute nicht so klar.

Teamsitzung 30. April 2012

Allgemeines

Projektwoche	11
Anwesende	cr, ms
Dauer	0.75h

Teamsitzung

- Diverse Diskussionen zur Teamarbeit

Iterations-Ende 01.05.2012

Allgemeines

Iteration	Construction 2
Dauer (ist)	87.50 Stunden
Dauer (soll)	103 Stunden

Verspäteter Abschluss der Iteration

Geplant war, diese Iteration am Freitag 20.4 abzuschliessen. Nach der Sitzung mit Jonathan Möller hatten wir sehr viel neuen Input: Wir wollten zuerst unsere Gedanken neu ordnen in einen Überblick haben, wie es weitergeht und wo das Projekt hinsteuert. Daher haben wir die Abschlussarbeiten dieser Iteration einige Tage verschoben.

Reflexion

Sitzung fyayc

Die Sitzung mit Jonathan Möller hatte eine spezielle Wirkung auf uns: Nach der Sitzung hatten wir beide das Gefühl, dass PSA so wie wir es geplant hatten, keinen Nutzen bringen würde. Wir verstanden beide Jonathan so, dass ein solches Programm nur sinnvoll sei, wenn es direkt zur Konfiguration eines PIM dienen könnte. Nach dem wir aber die Aufnahme der Sitzung nochmals durchgehört haben, verstanden wir Jonathans Input anders.

Unsere Idee

Aufgrund des Inputs von Jonathan entwickelten wir eine weitere Idee, wie in PSA die Produkttypen modelliert werden sollen (Präsentationsmatrix). Bei der Sitzung mit DK zeigte sich, dass diese Idee heute in der Praxis nicht verwendet werden würde.

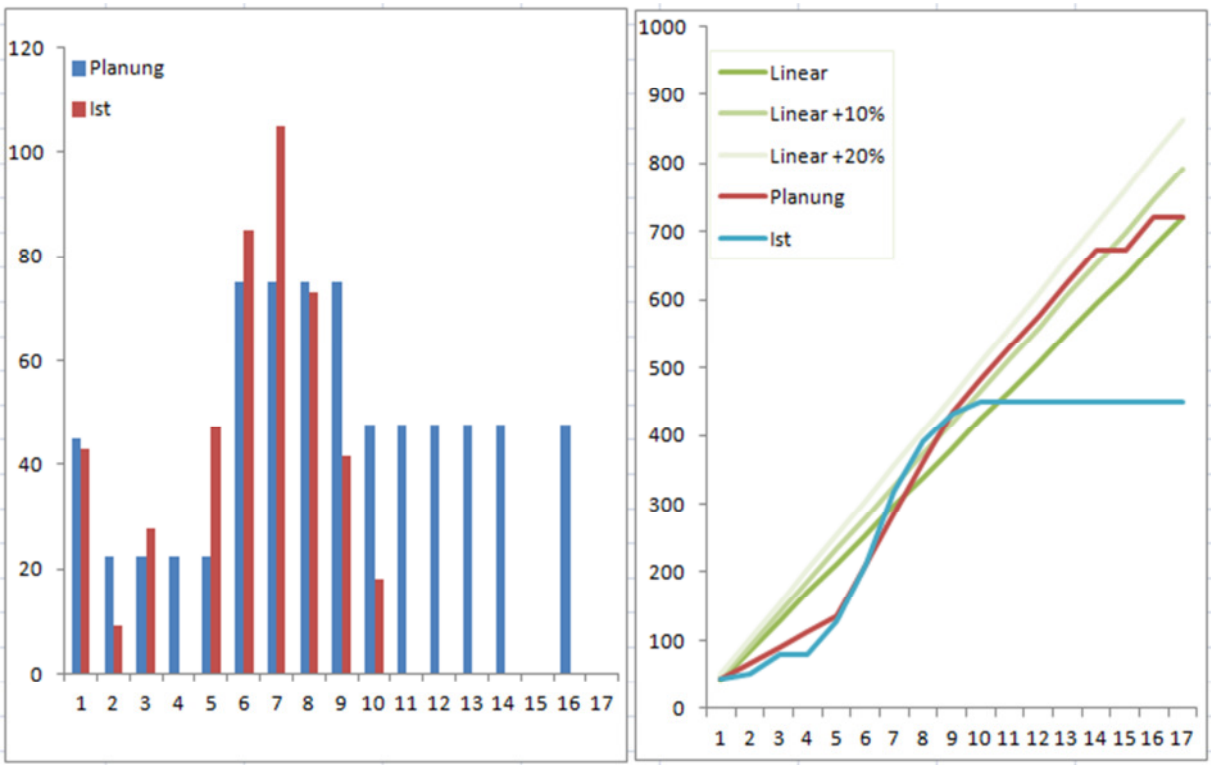
Hätte diese Idee funktioniert, wäre die Zeit sinnvoll investiert gewesen. Wir gehen auch davon aus, dass wir eigene Ideen entwickeln sollen. Somit muss man die verlorene Zeit für eine abgelehnte Idee auch in Kauf nehmen - auch wenn dies für die übrigbleibende Projektzeit schmerzhaft ist.

Input von DK 26.4

An der Sitzung vom 26.4 hat DK uns weiteren Input gegeben, wie die modellierungsarbeiten in der Praxis aussehen. Basierend auf diesen Ausführungen können wir uns nun neue Darstellungsvarianten für die Produkttyp Modellierung überlegen.

Rollen

Projekt Manager



Dank des grossen Efforts in den Wochen 7 und 8 konnten eine gemütlichere Woche 10 erlauben: Wir sind nur wenig hinter der gesamt Planung zurück. Die weitere Planung für das Programm ist allerdings eng gepackt. Um den neuen Input von DK zu verarbeiten werden wir wohl mindestens 10% mehr als die geforderten 720 Stunden leisten müssen.

Iterations-Start 29.04.2012

Allgemeines

Iteration Construction 3

Projektwochen 11

Datum 29.04.2012 bis 06.05.2012

Dauer (soll) 84 Stunden

Iterationsziele

Inhalt Um das Treffen mit fyayc vorzubereiten mussten wir die Construction 1 abbrechen. Construction 3 dient zum Abschliessen von Construction 1 (speziell das Refactoring) und zusätzlich eine weitere Alternativdarstellung zum UML-Editor zu finden.

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt	Zeitpunkte
112		Normal	CodeContracts Recherche		30.04.2012	04.05.2012	3.0		
111		Normal	Log4Net Lernen		30.04.2012	03.05.2012	5.0		
107		Normal	Doku C1 & C3		05.05.2012	06.05.2012	8.0		
106		Normal	GUI nicht neu aufbauen		05.05.2012	06.05.2012	6.0		
105		Normal	Debugging		04.05.2012	06.05.2012	8.0		
104		Normal	Code - Todos		01.05.2012	06.05.2012	8.0		
103		Normal	Unit-Test		01.05.2012	04.05.2012	8.0		
102		Normal	Konfigurationstab		01.05.2012	03.05.2012	5.0		
101		Normal	UML Editor		01.05.2012	03.05.2012	8.0		
100		Normal	Refactoring		30.04.2012	03.05.2012	12.0		
99		Normal	Planung Construction 3		30.04.2012	30.04.2012	1.0		
98		Normal	Sitzung DK 03.05		03.05.2012	03.05.2012	4.0		
97		Hoch	Alternative Darstellungen AIH überlegen		30.04.2012	03.05.2012	8.0		

Abbildung 8: Arbeitspakete Construction 3

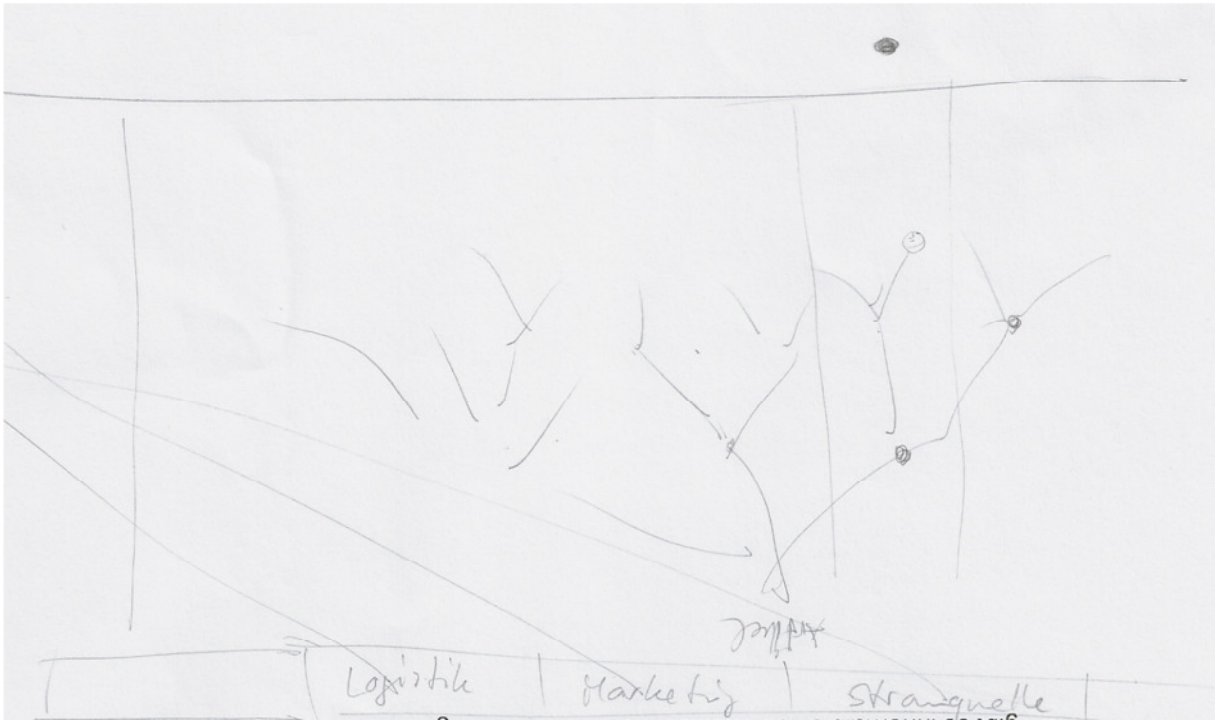
Sitzung 03.05.2012

Allgemeines

Projektwoche	11
Anwesende	cr, ms, dk
Dauer	1.5 min

Ablauf

DK erklärte uns den Ablauf der heutigen Präsentation für den Gegenleser (Thema erklären, Aufgabenstellung erläutern, Tool vorstellen und Probleme aufzeigen). Danach diskutierten wir weitere Darstellung und brachten die Idee von den Themen ein. Diese Idee sollte man umsetzten. Ob wir einen Abstrakten Produkttypen einführen ist noch unklar. Die Idee, dass wir die Vererbungshierarchie in eine Baumstruktur „drücken“ muss man genauer analysieren, wäre aber schön wenn es irgendwie möglich wäre.



Christoph Rosenberger
Michael Steiner

Iterations-Ende 10.05.2012

Allgemeines

Iteration	Construction 3
Dauer (ist)	95.5 Stunden
Dauer (soll)	95.5 Stunden

Vortrag mit Gegenleser, Neu Planung

Kurzfristig hatten wir noch einen Termin mit unserem Gegenleser Herrn Rudin. Somit mussten wir in dieser Iteration zusätzlich einen Vortrag vorbereiten und halten. Wir haben es in diesem Fall für besser befunden die Iteration zu verlängern, also die Timebox zu sprengen.

Erreichte und nicht erreichte Ziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
113		Normal	Treffen mit Gegenleser Hans Rudin		03.05.2012	03.05.2012	1.5	
112		Normal	CodeContracts Recherche		30.04.2012	04.05.2012	3.0	
111		Normal	Log4Net Lernen		30.04.2012	03.05.2012	5.0	
107		Normal	Doku C1 & C3		05.05.2012	06.05.2012	8.0	
106		Normal	GUI nicht neu aufbauen		05.05.2012	06.05.2012	6.0	
105		Normal	Debugging		04.05.2012	06.05.2012	8.0	
104		Normal	Code - Todos		01.05.2012	06.05.2012	8.0	
103		Normal	Unit-Test		01.05.2012	04.05.2012	8.0	
102		Normal	Konfigurationstab		01.05.2012	03.05.2012	5.0	
101		Normal	UML Editor		01.05.2012	03.05.2012	8.0	
100		Normal	Refactoring		30.04.2012	03.05.2012	12.0	
99		Normal	Planung Construction 3		30.04.2012	30.04.2012	1.0	
98		Normal	Sitzung DK 03.05		03.05.2012	03.05.2012	4.0	
97		Hoch	Alternative Darstellungen AIH überlegen		30.04.2012	03.05.2012	8.0	

Reflexion

GUI nicht neu aufbauen

Von Michael vor 11 Tagen hinzugefügt. Vor etwa 22 Stunden aktualisiert.

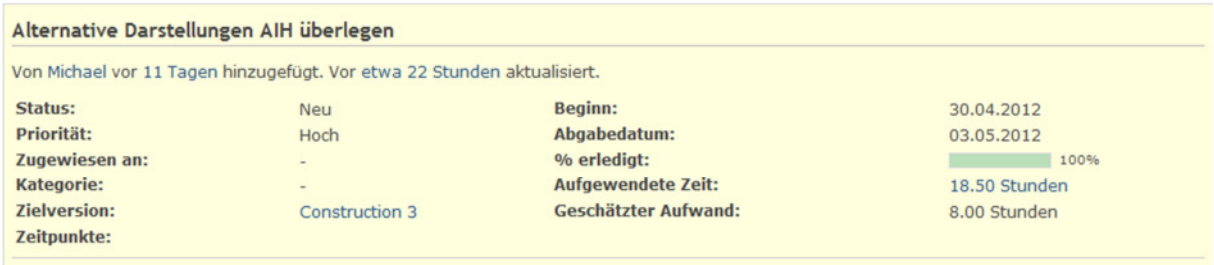
Status:	Neu	**Beginn:**	05.05.2012
Priorität:	Normal	**Abgabedatum:**	06.05.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	23.00 Stunden
Zielversion:	Construction 3	**Geschätzter Aufwand:**	6.00 Stunden
Zeitpunkte:			

Unteraufgaben Hinzufügen

Zugehörige Tickets Hinzufügen

Wir hatten verschiedene unerwartete Probleme:

- Beim Starten des Programms werden die Views in "umgekehrtert" Reihenfolge aufgebaut. Somit kann die aufrufende View das ViewModel noch nicht an die eingebetete View weitergeben. Schlussendlich haben wir eine pragmatische Lösung gewählt. (Im MAValueTemplateSelector wir geprüft, ob bereits ein ViewModel vorhanden ist. Solange kein ViewModel vorhanden ist, wird so oder so nichts auf dem Bildschirm angezeigt.)
- Wir wollten die Metaattribut Ansichten für AIHClass und AIHAttribute nicht CopyPasten. Die Idee die ViewModels als Generics zu implementieren stellte sich als untauglich heraus: In den XAML kann man Generics nicht ansprechen. Wir haben nun eine Lösung gewählt, in der sich unser ViewModel zweimal auf eine Message einschreibt.



Alternative Darstellungen AIH überlegen

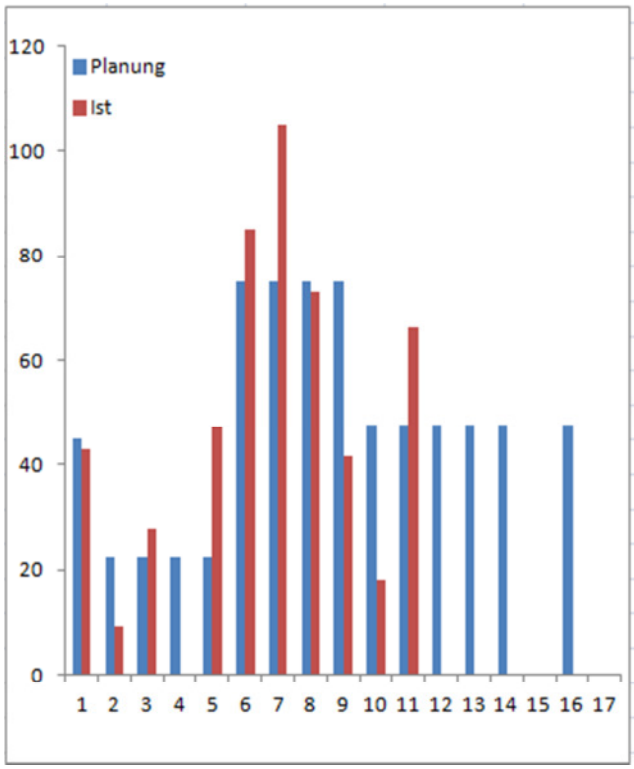
Von Michael vor 11 Tagen hinzugefügt. Vor etwa 22 Stunden aktualisiert.

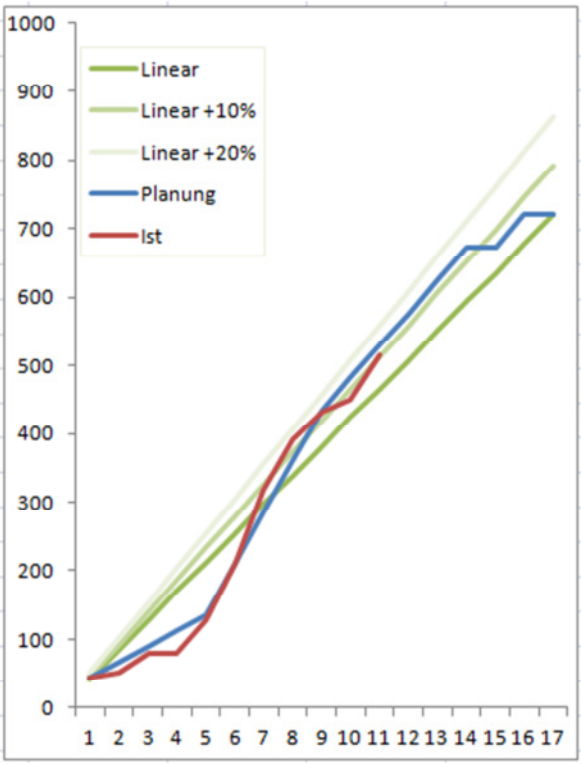
Status:	Neu	**Beginn:**	30.04.2012
Priorität:	Hoch	**Abgabedatum:**	03.05.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	18.50 Stunden
Zielversion:	Construction 3	**Geschätzter Aufwand:**	8.00 Stunden
Zeitpunkte:			

Hier war wohl die Schätzung schlicht zu optimistisch. Festzustellen bleibt höchstens, dass wir beim Designen für gewisse Ideen auch schon die technische Lösung kurz recherchiert haben. Wir wollten sicherstellen, dass wir nicht sehr ungewöhnliche Elemente in der View planen, welche wir nur schwer hätten umsetzen können.

Rollen

Projekt Manager





Sitzung DK 10.05.2012

Allgemeines

Projektwoche	12
Anwesende	cr, ms, dk
Dauer	1.25 h

Bachelor Prüfung

- Anwesende Personen: Hans Rudin (Gegenleser), Rudolf Mattmann (Experte), Daniel Keller (Betreuer)
- Die Bachelorprüfung geht ca. 1h. Wahrscheinlich 30min Vortrag und 30min Fragen
- Der Vortrag soll das ganze Projekt abdecken.
- Ein wesentliches Ziel der Prüfung ist zu sehen, dass wir das Projekt auch wirklich selbst umgesetzt haben.
- Poster richtet sich an jemand, der keine Ahnung vom ganzen Thema hat. Es muss also unbedingt auch erklärt werden, was ein PIM ist.

Themen

- Die Unterteilung in Themen ist ein sehr guter Input
- Mögliche Aspekte sind: Produkt, ERP, Logistik, Community Content, Interne Statistiken und Marketing Informationen
- Unser Wireframe für die Modellierung der Themen wurde von DK für gut befunden.

Sonstiges

- Ausblick: Wie könnte es weiter gehen: Zusammenhänge zwischen den Produkten (Upgrade / Downgrade, Zubehör, ...). Wobei dies wahrscheinlich nicht auf Produkttypen modelliert werden kann, sondern auf den Artikeln.
- Statistiken: Wie viele Produkttypen erben von welcher Klasse (wie viele Produkttypen sind zerbrechlich, wie viele sind in Briefform lieferbar, ...)
- Gesamtbild in 3D (oben Produkttypen -> Abstrakte Produkttypen -> Themen)

Iterations-Start Construction 4

Allgemeines

Iteration	Construction 4
Projektwochen	12 bis 13
Datum	10.05.2012 bis 17.05.2012
Dauer (soll)	82 Stunden

Feature Freeze

Unsere Ursprüngliche Planung hätte den Feature Freeze Ende Projektwoche 13 gehabt. Wir setzen die erste unserer zwei Reserve Wochen ein und verschieben den Feature Freeze entsprechend.

Planung bis Projektende

Iteration	Tag	Feature
Construction 4	Freitag 11.5	Suche Attribute
	Samstag 12.5	Themen modellieren
	Sonntag 13.5	Themen modellieren
	Dienstag 15.5	Themen modellieren
	Donnerstag 17.5	Produkttypeditor Wireframe
Construction 5	Freitag 18.5	Bilder, Konfig Tab, Codepflege (weitere Unittests und Refactorings)
	Samstag 19.5	Metadaten Strukturierung
	Sonntag 20.5	Enumertionen modellieren
Construction 6	Donnerstag 24.5	Produkttyp Editor
	Freitag 24.5	Produkttyp Editor
	Samstag 26.5	Eingabemasken Ansicht
	Sonntag 27. 5	Komplexe Metaattribute
Construction 7	Montag 28.5	Code Abgabe
	Dienstag 19.5	Code Abgabe
	Donnerstag 31.5	Code Abgabe
	Freitag 1.6	Code Abgabe
Transition	Ganze Woche 16	Dokumentation
	Ganze Woche 17	Dokumentation

Dies ist eine optimistische Schätzung. Weitere technische Probleme (z.B. mit WPF) sind nicht eingeplant. Auch haben wir keine Reserve mehr. Für einen Einsatz absolut zwingend sind allerdings nur die Pakete welche bis zum Sonntag den 20.5 geplant sind.

Iterationsziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand
122		Normal	Diverses		10.05.2012	17.05.2012	8.0
121		Normal	Produkttypeditor Wireframe		17.05.2012	17.05.2012	8.0
116		Normal	Themen modellieren		12.05.2012	14.05.2012	48.0
117	Feature #116	Normal	Recherche Tree in Datagrid		12.05.2012	12.05.2012	8.0
118	Feature #116	Normal	Recherche Drag & Drop		12.05.2012	12.05.2012	8.0
119	Feature #116	Normal	View aufbauen		13.05.2012	13.05.2012	8.0
120	Feature #116	Normal	Logik programmieren		13.05.2012	14.05.2012	24.0
115		Normal	Attribut Suche		11.05.2012	11.05.2012	16.0

Iterations-Ende 19.05.2012

Allgemeines

Iteration	Construction 4
Dauer (ist)	106.75 Stunden
Dauer (soll)	82 Stunden

UInt2

Wir erhielten diese Woche das Feedback von M. Stolze für unser UInt2 Projekt und mussten unser Dokument vor der Abgabe noch einmal überarbeiten. Dies kostete uns einen Tag arbeit.

Erreichte und nicht erreichte Ziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
124		Normal	Planung Construction 4		11.05.2012	11.05.2012		
122		Normal	Diverses		10.05.2012	17.05.2012	8.0	
121		Normal	Produkttypeditor Wireframe		17.05.2012	17.05.2012	8.0	
116		Normal	Themen modellieren		12.05.2012	14.05.2012	48.0	
117	Feature #116	Normal	Recherche Tree in Datagrid		12.05.2012	12.05.2012	8.0	
118	Feature #116	Normal	Recherche Drag & Drop		12.05.2012	12.05.2012	8.0	
119	Feature #116	Normal	View aufbauen		13.05.2012	13.05.2012	8.0	
120	Feature #116	Normal	Logik programmieren		13.05.2012	14.05.2012	24.0	
115		Normal	Attribut Suche		11.05.2012	11.05.2012	16.0	

Produkttypeditor

Wir haben festgestellt, dass wir keinen Produkttypeditor benötigen, sondern ihn direkt in der Themenmodellierung integrieren können.

Reflexion

Attribut Suche

Attribut Suche

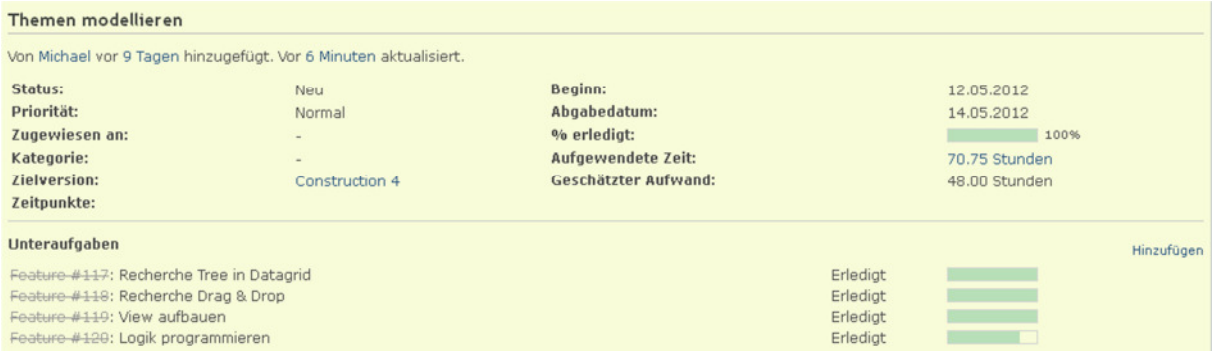
Von Michael vor 9 Tagen hinzugefügt. Vor 7 Minuten aktualisiert.

Status:	Erledigt	**Beginn:**	11.05.2012
Priorität:	Normal	**Abgabedatum:**	11.05.2012
Zugewiesen an:	-	**% erledigt:**	100%
Kategorie:	-	**Aufgewendete Zeit:**	30.00 Stunden
Zielversion:	Construction 4	**Geschätzter Aufwand:**	16.00 Stunden
Zeitpunkte:			

Christoph Rosenberger
Michael Steiner

Diverse technische sowie design Probleme (ValueConverter, Anzeigen der korrekten Input-Felder, …) kumuliert ergeben einen Aufwand, der fast doppelt so hoch ist, wie geschätzt. Wir waren schlicht zu optimistisch.

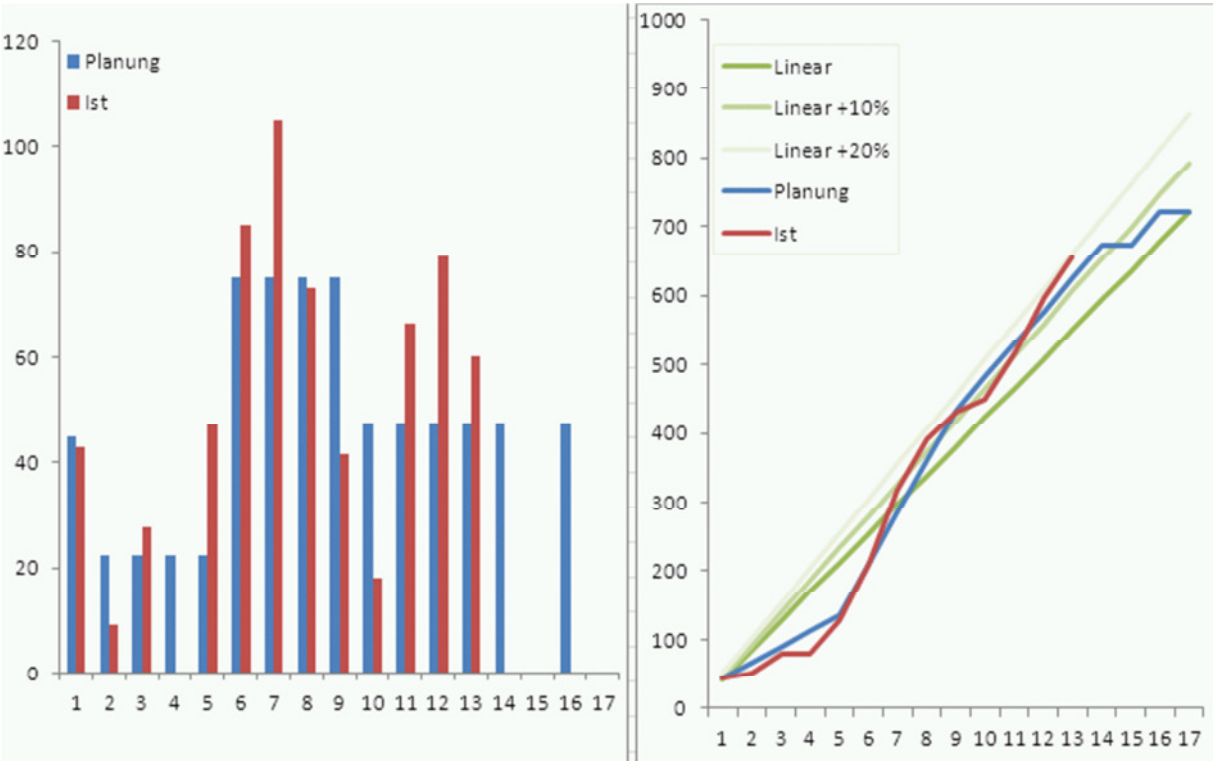
Themen modellieren



Die Idee, dass man in einer Treeview noch ein Datagrid integriert, existiert in WPF nicht. Dies führte dazu, dass wir dies selber umsetzten mussten. Dies führte zu diversen Problemen (Selection musste selber implementiert werden, Breite der Spalten mussten manuell nachgeführt werden, …)

Rollen

Projekt Manager



Iterations-Start Construction 5

Allgemeines

Iteration	Construction 5
Projektwochen	13
Datum	19.05.2012 bis 20.05.2012
Dauer (soll)	28 Stunden

Planung bis Projektende

Iteration	Tag	Feature
Construction 5	Samstag 19.5	Themeneditor
	Sonntag 20.5	Themeneditor
Construction 6	Donnerstag 24.5	Enumertionen modellieren
	Freitag 24.5	Metadaten Strukturierung
	Samstag 26.5	Bilder, Konfig Tab, Codepflege (weitere Unittests und Refactorings)
	Sonntag 27. 5	Eingabemasken Ansicht
Construction 7	Montag 28.5	Code Abgabe
	Dienstag 19.5	Code Abgabe
	Donnerstag 31.5	Code Abgabe
	Freitag 1.6	Code Abgabe
Transition	Ganze Woche 16	Dokumentation
	Ganze Woche 17	Dokumentation

Für folgende Features wird die Zeit vermutlich nicht reichen:

- Komplexe Metaattribute
- Filtern & Markieren in Tree

Iterationsziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand
127		Normal	Tab Vererbungshierarchie		19.05.2012	19.05.2012	22.0
128	Feature #127	Normal	Anzeige für Attribute, Klassen, Themen		19.05.2012	19.05.2012	3.0
129	Feature #127	Normal	Drag & Drop beenden (View & ViewModels)		19.05.2012	19.05.2012	3.0
130	Feature #127	Normal	Logik (DataAcces und ViewModels anpassen)		19.05.2012	19.05.2012	6.0
131	Feature #127	Normal	Reihenfolge der Themen und Klassen		19.05.2012	19.05.2012	3.0
132	Feature #127	Normal	Mehrfachvererbung anzeigen + modellieren		19.05.2012	19.05.2012	4.0
133	Feature #127	Normal	Icons erstellen		19.05.2012	19.05.2012	1.0
134	Feature #127	Normal	Icons einbinden		19.05.2012	19.05.2012	2.0
126		Normal	Datenstrukturierung Tabs für Klassen und Themen		19.05.2012	19.05.2012	4.0
125		Normal	Planung Construction 5		19.05.2012	19.05.2012	2.0

Iterations-Ende 20.05.2012

Allgemeines

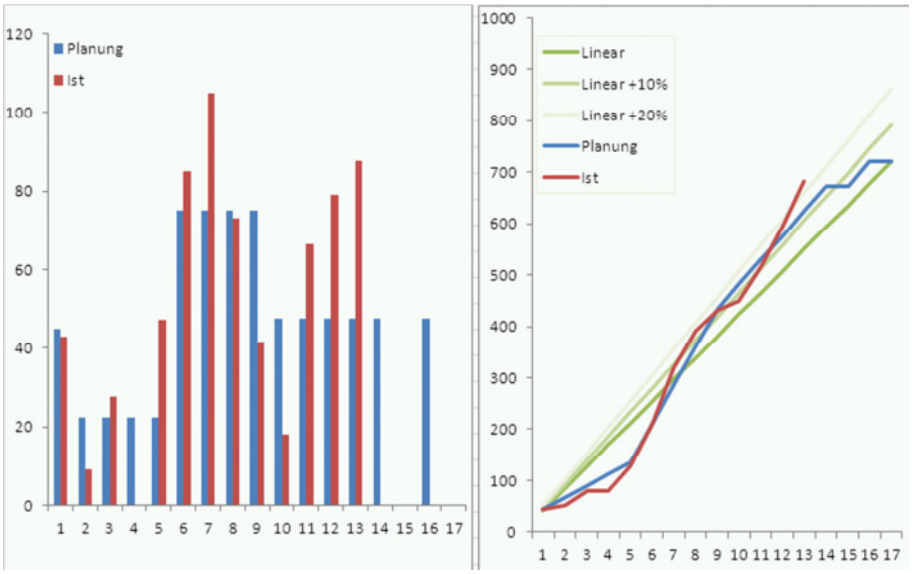
Iteration	Construction 5
Dauer (ist)	27.25 Stunden
Dauer (soll)	28 Stunden

Erreichte und nicht erreichte Ziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
127		Normal	Tab Vererbungshierarchie		19.05.2012	19.05.2012	22.0	
128	Feature #127	Normal	Anzeige für Attribute, Klassen, Themen		19.05.2012	19.05.2012	3.0	
129	Feature #127	Normal	Drag & Drop beenden (View & ViewModels)		19.05.2012	19.05.2012	3.0	
130	Feature #127	Normal	Logik (DataAcces und ViewModels anpassen)		19.05.2012	19.05.2012	6.0	
131	Feature #127	Normal	Reihenfolge der Themen und Klassen		19.05.2012	19.05.2012	3.0	
132	Feature #127	Normal	Mehrfachvererbung anzeigen + modellieren		19.05.2012	19.05.2012	4.0	
133	Feature #127	Normal	Icons erstellen		19.05.2012	19.05.2012	1.0	
134	Feature #127	Normal	Icons einbinden		19.05.2012	19.05.2012	2.0	
126		Normal	Datenstrukturierung Tabs für Klassen und Themen		19.05.2012	19.05.2012	4.0	
125		Normal	Planung Construction 5		19.05.2012	19.05.2012	2.0	

Reflexion

Es gab in dieser Iteration keine Überraschungen weshalb wir unsere Ziele auch in der geplanten Zeit erreichen konnten.



Christoph Rosenberger
Michael Steiner

Iterations-Start Construction 6

Allgemeines

Iteration	Construction 6
Projektwochen	13 bis 14
Datum	20.05.2012
Dauer (soll)	57 Stunden

Abwesenheit CR

Es ist unklar, wie viel Christoph Rosenberger diese Woche arbeiten kann, da er seinen Leistenbruch operieren lassen muss.

Planung bis Projektende

Iteration	Tag	Feature
Construction 6	Donnerstag 24.5	Enumertionen modellieren
	Freitag 24.5	Metadaten Strukturierung
	Samstag 26.5	Konfig Tab
	Sonntag 27. 5	Eingabemasken Ansicht
Construction 7	Montag 28.5	Code Abgabe
	Dienstag 19.5	Code Abgabe
	Donnerstag 31.5	Code Abgabe
	Freitag 1.6	Code Abgabe
Transition	Ganze Woche 16	Dokumentation
	Ganze Woche 17	Dokumentation

Für folgende Features wird die Zeit vermutlich nicht reichen:

- Komplexe Metaattribute
- Filtern & Markieren in Tree

Iterations-Ende 24.05.2012

Allgemeines

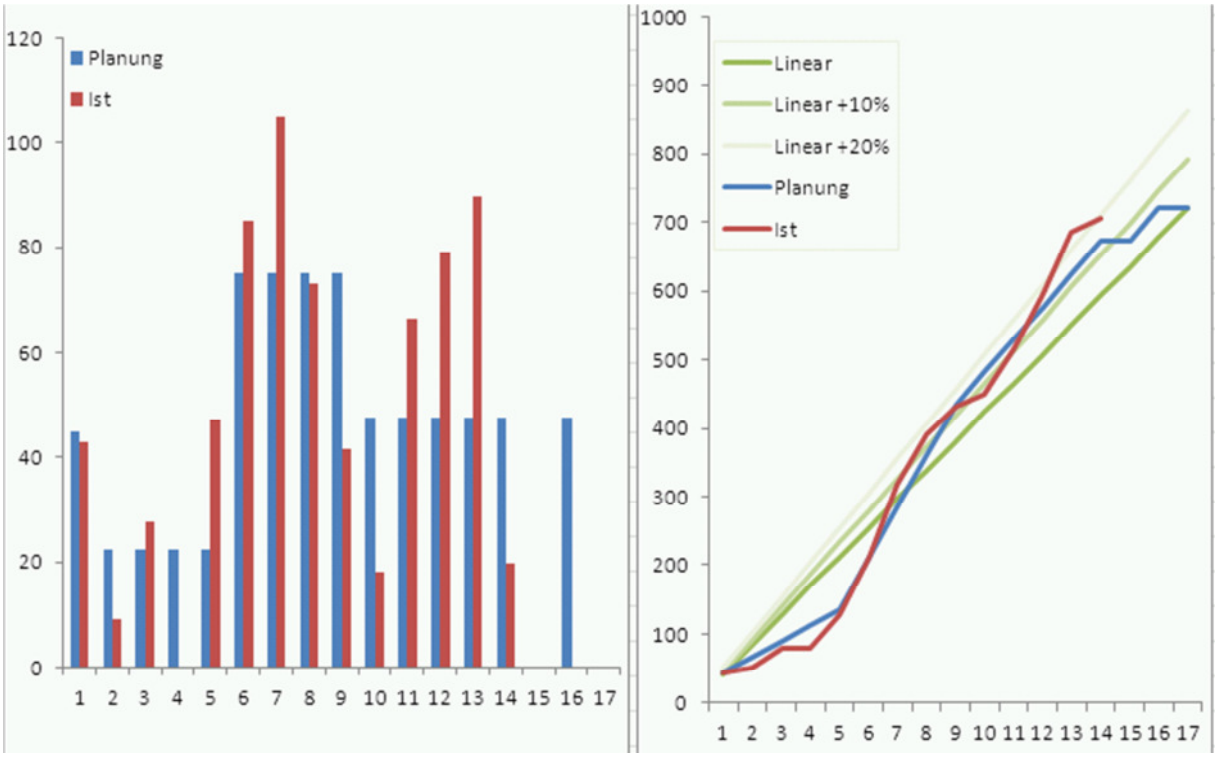
Iteration	Construction 6
Dauer (ist)	21.75 Stunden
Dauer (soll)	57 Stunden

Erreichte und nicht erreichte Ziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
143		Normal	Sitzung DK 24.05		24.05.2012	24.05.2012	2.0	
142		Normal	Planung Construction 6		20.05.2012	20.05.2012	2.0	
138		Normal	Eingabemasken Ansicht		21.05.2012	27.05.2012	12.0	
137		Normal	Konfigurationstab		21.05.2012	27.05.2012	17.0	
139	Feature #137	Normal	PSA Konfiguration (Logo setzen)		21.05.2012	27.05.2012	1.0	
140	Feature #137	Normal	Backup		21.05.2012	27.05.2012	8.0	
141	Feature #137	Normal	Export		21.05.2012	27.05.2012	8.0	
136		Normal	Metadaten Strukturierung		21.05.2012	27.05.2012	14.0	
135		Normal	Enumertionen modellieren		21.05.2012	27.05.2012	10.0	

Wir schliessen diese Iteration ab um die Prioritäten anhand des neuen Input von DK zu setzen.

Reflexion



Christoph Rosenberger
Michael Steiner

Sitzung DK 24.05.2012

Allgemeines

Projektwoche	15
Anwesende	ms, dk
Dauer	0.75h

Fragen

- Wir haben noch 4 Tage, welche wir planen können. Wie sollen wir unsere Prioritäten setzen? Was wünschen sie noch für Features?
- Wann können wir die Präsentation halten?

Neuer Input

- Attribute importieren
  - Da wir keine Beispieldaten bekommen, können wir von einem Textfile ausgehen, welches pro Zeile ein Attributname enthält (keine Metadaten).
- Daten Export
  - Es wäre schön, wenn man den Baum als xml exportieren könnte. Die Mehrfachvererbung kann ohne weiteres weggelassen werden.
- Ähnliche Attribute finden
  - Gut wäre, wenn man ähnliche Attribute findet, damit man Duplikate vermeiden kann. Ev. helfen folgende Stichworte: Trigram, Soundex
- Komplexe Metaattribute
  - Die komplexen Metaattribute können weggelassen werden
- Eingabemasken
  - Sind immer noch wichtig

Christoph Rosenberger
Michael Steiner

Iterations-Start Construction 7

Allgemeines

Iteration	Construction 7
Projektwochen	14 bis 15
Datum	24.05.2012
Dauer (soll)	52 Stunden

Planung bis Projektende

Iteration	Tag	Feature
Construction 7	Donnerstag 24.5	Eingabemasken Ansicht Recherche Duplikaten Finder
	Freitag 25.5	Konfig Tab (Import, Export)
	Samstag 26.5	Metadaten Strukturierung
	Sonntag 27. 5	Duplikaten Finder
Transition	Montag 28.5	Code Abgabe
	Dienstag 19.5	Code Abgabe
	Donnerstag 31.5	Code Abgabe
	Freitag 1.6	Code Abgabe
	Ganze Woche 16	Dokumentation
	Ganze Woche 17	Dokumentation

Für folgende Features wird die Zeit vermutlich nicht reichen:

- Komplexe Metaattribute
- Filtern & Markieren in Tree

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
149		Normal	Weitere Filter (insbesondere Dublikaten Filter)		27.05.2012	27.05.2012	16.0	
148		Normal	Metadaten Strukturierung		26.05.2012	26.05.2012	10.0	
147		Normal	Eingabemaskenansicht		24.05.2012	24.05.2012	12.0	
146		Normal	Export		25.05.2012	25.05.2012	8.0	
145		Normal	Import		25.05.2012	25.05.2012	4.0	
144		Normal	Planung Construction 7		24.05.2012	24.05.2012	2.0	

Christoph Rosenberger
Michael Steiner

Iterations-Ende 29.05.2012

Allgemeines

Iteration	Construction 7
Dauer (ist)	64 Stunden
Dauer (soll)	52 Stunden

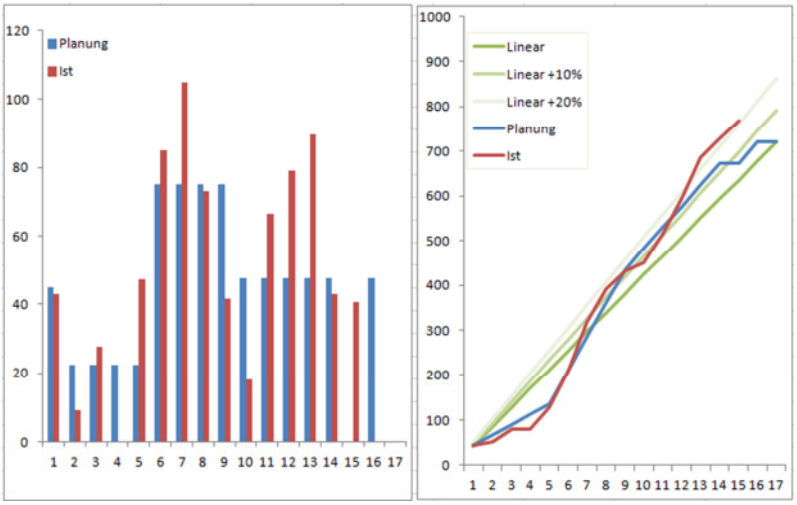
Einschub Performanceoptimierung

Als der Import funktionierte, hatten wir auch zum ersten Mal viele Daten im Programm. Wir mussten leider feststellen, dass unsere Performance schlechter als erwartet war, weshalb wir hier einen kleinen Einschub machten und dieses Problem behoben. Wir verschoben die Fertigstellung der Features, denn wenn wir die Performance nicht verbessert hätten, könnte man das Programm nicht gebrauchen.

Erreichte und nicht erreichte Ziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand	% erledigt
150		Sofort	Performance verbessern		25.05.2012	27.05.2012	16.0	
149		Normal	Weitere Filter (insbesondere Dublikaten Filter)		27.05.2012	27.05.2012	16.0	
148		Normal	Metadaten Strukturierung		26.05.2012	26.05.2012	10.0	
147		Normal	Eingabemaskenansicht		24.05.2012	24.05.2012	12.0	
146		Normal	Export		25.05.2012	25.05.2012	8.0	
145		Normal	Import		25.05.2012	25.05.2012	4.0	
144		Normal	Planung Construction 7		24.05.2012	24.05.2012	2.0	

Reflexion



Anhang: Projektverlauf

Christoph Rosenberger
Michael Steiner

Start Transition 1: Code Abgabe

Allgemeines

Iteration	Transition 1
Projektwochen	15
Datum	29.5 bis 3.6
Dauer (soll)	33 Stunden

Planung bis Projektende

Transition 1	Dienstag 29.5	Code Abgabe
	Donnerstag 31.5	Code Abgabe
	Freitag 1.6	Code Abgabe
	Samstag 2.6	
	Sonntag 3.6	
Transition 2	Ganze Woche 16	Dokumentation
	Ganze Woche 17	Dokumentation

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand
154		Normal	Sitzung DK 31.05		31.05.2012	31.05.2012	4.0
153		Normal	Todos im Code		29.05.2012	01.06.2012	3.0
152		Normal	Debugen		29.05.2012	01.06.2012	16.0
151		Normal	Tests entwickeln und durchführen		29.05.2012	01.06.2012	10.0

Sitzung DK 31.05.2012

Allgemeines

Projektwoche 15

Anwesende cr, ms, dk

Dauer 2h

Bachelor Prüfung

- Präsentation was hat ihnen im SE2 Projekt nicht gefallen?
- Unterschrift Aufgabenstellung
- Programm präsentieren, Details klären
    - Sind die Datentypen die wir unterstützen gut? Benötigen sie noch einen? → Ja Zeit, Datum getrennt
    - Masken schöner gestalten
    - XML-Export ist so gut
- Dokumentation

Christoph Rosenberger
Michael Steiner

End Transition 1 Code Abgabe

Allgemeines

Iteration	Transition 1
Dauer (ist)	38 Stunden
Dauer (soll)	33 Stunden

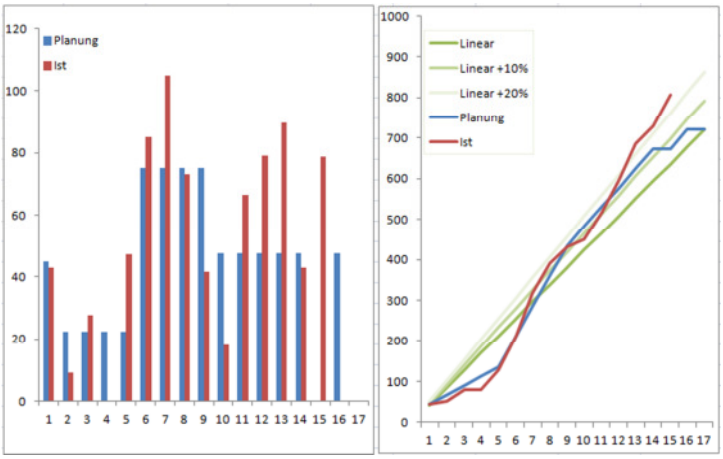
UML Generator

Eigentlich wäre der Feature Freeze bereits gewesen. Beim Aufräumen des Programms hatten wir eine Diskussion über den UML Editor. Die ursprüngliche Planung war mit einem Warnhinweis darauf hinzuweisen, dass er nicht gebraucht werden soll. Er zeigte auch einfach alle Klassen oben links in der Ecke. Dies fanden wir aber sehr unschön, weil der Rest des Programms super ist, wäre dies irgendwie störend gewesen. Den ganzen Editor löschen wollten wir auch nicht. Eine kurze Analyse ergab, dass wir nur die Positionen berechnen mussten um einen UML Generator zu erstellen. Mit einem einfachen Algorithmus war dies nicht schwierig. So entschieden wir uns, noch ein letztes (cooles) Feature zu implementieren.

Erreichte und nicht erreichte Ziele

#	Übergeordnete Aufgabe	Priorität	Thema	Zugewiesen an	Beginn	Abgabedatum	Geschätzter Aufwand
155		Normal	UML Generieren		30.05.2012	30.05.2012	2.0
154		Normal	Sitzung DK 31.05		31.05.2012	31.05.2012	4.0
153		Normal	Todos im Code		29.05.2012	01.06.2012	3.0
152		Normal	Debugen		29.05.2012	01.06.2012	16.0
151		Normal	Tests entwickeln und durchführen		29.05.2012	01.06.2012	10.0

Projekt Manager



Christoph Rosenberger
Michael Steiner

Start Transition 2 Doku

Allgemeines

Iteration	Transition 2
Projektwochen	16 bis 17
Datum	04.06.2012 bis 15.06.2012
Dauer (soll)	? Stunden

Dokumentation

Diese letzten zwei Wochen bleiben für die Dokumentation. Unser Budgetziel von insgesamt 864 Stunden werden wir kaum einhalten können: 58 Stunden werden wohl kaum genug sein. Wir verzichten aber darauf eine Zeitschätzung zu erstellen. Ohne sinnvolle, kleine Arbeitspakete können wir keine Sinnvolle Schätzung machen. Und genug Erfahrung haben wir auch nicht.

Michael hat am 14.06.2012 seine Masteraufnahmeprüfung und wird sich darauf vorbereiten müssen.

Anhang E6
User Interface 2 - Miniprojekt

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

Visualisierung von komplexen Produktstrukturen

Userinterface 2 - Miniprojekt

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2012

Autor(en):	Christoph Rosenberger, Michael Steiner
Betreuer:	Dr. Markus Stolze
Projektpartner:	foryouandyourcustomers, Pfäffikon ZH
Ansprechpartner:	Dr. Daniel Keller

Inhaltsverzeichnis

Christoph Rosenberger
Michael Steiner

1 Einleitung

Kontext

Dieses UInt2 Miniprojekt setzen wir im Kontext unserer Bachelorarbeit um. Unser Thema ist die " Visualisierung von komplexen Produktstrukturen".

Das Ziel unserer Arbeit ist in erster Linie die Entwicklung von Ideen und neuen Konzepten. Die Umsetzung als Software (PSA, Produkt Structure Architect) dient primär der Konkretisierung und um eine Diskussionsgrundlage zu schaffen.

Im Zusammenhang mit dem UInt2 Miniprojekt möchten wir eine frühe Version von PSA planen. Für die ersten Ideen und Diskussionen brauchen wir auch keine funktionierende Software, hierzu genügen einfache Papierprototypen.

Workproducts

Folgende Workproducts haben wir mit Herrn Stolze für unser Miniprojekt vereinbart:

- 3 Sollszenarien
- Diese Szenarien bis zu drei Mal mit Herrn Keller überarbeiten
- 2 Papierprototypen
- Diese mit Herrn Keller testen
- Überarbeitete Papierprototypen

Aufbau des Dokumentes

In den Kapiteln 3 und 4 zeigen wir die Entstehung der Resultate auf. Hierzu sind jeweils diejenigen Versionen welche wir mit dem User D. Keller besprochen und validiert haben festgehalten. Im Kapitel „2 Resultat“ haben wir das Ergebnis, welches wir mit D. Keller erarbeitet und anhand des Feedbacks von M. Stolze überarbeitet haben festgehalten.

Alle Texte an welchen wir Änderungen vorgenommen haben, kommen somit in späteren Iterationen nochmals (überarbeitet) vor.

Im letzten Kapitel „Nachtrag“ haben wir einige Screenshots des Programms um zu zeigen, wie diese Arbeit in unser Programm eingeflossen ist.

2 Resultat

In diesem Kapitel haben wir die Resultate unsres Projekts nach Garrett dokumentiert.

2.1 Strategie

2.1.1 Heutige Arbeitsweise

Heutige Arbeitsweise

Eine Produktstrukturierung könnte heute nach folgenden Schritten stattfinden:

- Die heutige IT-Landkarte wird aufgezeigt und bestimmt welche Systeme beibehalten, welche verändert und welche neu gemacht werden sollen
- Es werden Ziele festgelegt z.B.:
  - Bessere Usability
  - Weniger Retouren
  - Schnellere Lieferungen
  - Bessere Kundenbindung
- Es werden Einschränkungen festgelegt z.B.:
  - Daten die unter Datenschutz stehen
  - Technologieeinschränkungen
- Requirements und UseCases
- Evaluation des Implementationspartner
- Es wird festgelegt welche Systeme neu gemacht werden sollen, und welche bestehen bleiben
- Sitzungen für gemeinsames Verständnis von Begriffen (wobei die nicht sauber und exakt definiert werden)
- Der Kunde erstellt eine Liste von Attributen (ohne weitere Definitionen) in Excel. (→ Hier wird PSA ansetzen)
- Vorgehen wird besprochen
  - Iterationen planen (1-2 Wochen für fyayc Entwickler)
  - Meilensteine für Kunden, alle 3 Monate ein produktives Release, z.B. nur in einzelnen Ländern
- In Iterationen werden die Systeme von fyayc konfiguriert: PIM, CRM, MAM, ERP, Webshop, Supportsystem, etc.

Arbeitsmaterialien

- Präsentationsmittel (Eigene, vom Kunden, vom Lieferanten):
  - Katalog
  - Webshop
  - Apps
- Bestehende Systeme
  - CRM, etc.
- Dokumentation vom PIM-Programm
  - Attributliste von Riversand

Probleme

- Die Arbeiten finden unstrukturiert statt. Weder gibt es eine einheitliche Vorgehensweise, noch werden die Resultate sauber strukturiert dokumentiert.
- Verschiedene Begriffe sind nicht sauber definiert und werden für verschiedene Konzepte eingesetzt. Im Gegenzug gibt es für einige Konzepte auch keine Begriffe.

2.1.2 Einsatzkontext von PSA

Einsatz während Projektablauf

Im obigen Ablauf (Heutige Arbeitsweise) findet PSA ab der Erstellung der Attributliste Verwendung.

Benutzer

Für die Modellierung wird PSA von fyayc Mitarbeitern verwendet. Um dem Kunden die Resultate der Produktstrukturmodellierung aufzuzeigen und diese zu besprechen, kann es nützlich sein, diese Resultate zu visualisieren.

fyayc-Benutzer

- Er hat ein überdurchschnittliches logisch-abstraktes Denkvermögen.
- Er weiss wie man eine Produktstrukturierung durchführt und wozu.
- Er hat eine (kurze) Einführung in PSA erhalten und kennt die verwendeten Begriffe.

Kundenmitarbeiter

Bei Darstellungen welche für Diskussionen mit dem Kunden gedacht sind, darf von keinen spezifischen Kenntnissen ausgegangen werden.

Ziel

Folgende Ziele sollen durch den Einsatz von PSA einfacher erreicht werden:

- Begriffsdefinitionen finden (zusammen mit Kunden)
- Finden aller Attribute
- Konsistente Namensgebung der Attribute
- Zusammenhang zwischen den Produkten aufzeigen
  - Zubehör
  - Bestandteil (Composite)
  - Variation
  - alternativ Produkte
- Verständnis der Produktstruktur verbessern. Strukturierung des Mindmodels.
- Vorarbeit für Erstellung des logischen Domainmodells
  - Attribute
  - Datentypen
  - wie Vererbung Umsetzen
- Hilfestellung für Erstellung der Navigationshierarchie
- Hilfestellung für Erstellung Faceted Search
- Hilfestellung für Auswertungen
  - Z.B. kann man im laufenden Betrieb Absatzstatistiken erstellen
- Vereinfachtes Hinzufügen neuer Produkte (im laufenden Betrieb) (Man muss nicht mehr alle nötigen Attribute angeben sondern muss nur noch sagen, von welchen Klassen geerbt werden soll.)
- Die Definition von Eingabemasken wird unterstützt oder sogar automatisiert (Z.B. kann anhand der Metaattribute eine Eingabemaske für einen spezifischen Mitarbeiter generiert werden, welchem nur diejenigen Attribute angezeigt werden, welche für ihn auch relevant sind).

2.1.3 Szenarios

Szenario 1: Neues Attribut erfassen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Dagitec verkauft Computerzubehör und als Spezial Nische Raritäten aus der Autowelt über einen Webshop. Die Produktstrukturierung ist schon recht weit fortgeschritten, nächste Woche findet das vierte Meeting mit Dagitec statt.

Kevin liest in der Zeitung, dass der Ford T jetzt auch in Pink erhältlich sei. Da er sich nicht sicher ist, ob das aktuelle Datenmodell diese Konfiguration zulässt, startet er die PSA-Applikation und überprüft ob es ein Attribut „Farbe“ für dieses Modell gibt. Dazu öffnet er das Suchfenster und testet mehrere mögliche Synonyme. Tatsächlich scheint kein exakt passendes Attribut zu existieren. Kevin erfasst das neue Attribut im System und erfasst auch gleich die Metaattribute:

- Attributtyp: String
- Muss übersetzt werden: ja

Damit dieses spezielle Farbattribut (mit dem Range Schwarz, Pink) nicht ausversehen für ein anderes Fahrzeug verwendet wird, erfasst Kevin noch ein weiteres Metaattribut “Hinweis”.

Zufrieden fügt er das neue Attribut in der Klasse Auto ein.

http://de.wikipedia.org/wiki/Ford_Modell_T

Szenario 2: Attribut Suchen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Dagitec verkauft Computerzubehör und als Spezial Nische Raritäten aus der Autowelt über einen Webshop. Die Produktstrukturierung ist noch nicht weit fortgeschritten nächste Woche findet das zweite Meeting mit Dagitec statt.

Kevin studiert mit einem Kunden dessen Produktkatalog und erfasst mit ihm eine neue Klasse welche die Computergehäuse repräsentiert. Beim Einfügen des neuen Attributes Länge haben die Beiden verschiedenen Attributnamen für dasselbe Attribut im Kopf. Kevin möchte es “Innenlänge” nennen, da es ihm der Kunde auch so erklärt hat. Der Kunde findet aber “Länge” passender, denn niemand würde in der Praxis von “Innenlänge” sprechen. Um zu überprüfen, ob es schon ein solches Attribut mit der gewünschten Bedeutung gibt, wechseln sie auf die Suchmaske und suchen nach den beiden Namen. Tatsächlich existieren beide Attribute. Um deren Bedeutung genauer zu verstehen, schauen sie sich die erfassten Metaattribute an. Die beiden Attribute scheinen auf den ersten Blick sehr ähnlich und sie werden sich nicht einig, welches Attribut sie jetzt verwenden sollen. Darauf ruft Kevin beim Erfasser dieser beiden Attribute an und fragt nach deren genauen Bedeutung. Jetzt weiss er, dass er das Attribut “Innenlänge” verwenden muss. Er notiert seine Erkenntnisse sofort im

Kommentarfeld des Attributes.

Szenario 3: Neue Klasse erstellen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Dagitec verkauft Computerzubehör und als Spezial Nische Raritäten aus der Autowelt über einen Webshop. Die Produktstrukturierung ist schon recht weit fortgeschritten nächste Woche findet das dritte Meeting mit Dagitec statt.

Rechtzeitig zur Osterzeit bringt Dagitec einen neuen Katalog mit top aktuellen Produkten auf den Markt. K.H. von der Firma Dagitec schickt die neue Auflage an Kevin. Am nächsten Tag blättert Kevin den neuen Katalog durch. Darin findet er ein neues Produkt: Eine solarbetriebene Fotokamera. Zunächst begeistert von dieser Innovation, bemerkt er anschliessend aber, dass man zu diesem Gerät kein Netzteil kaufen kann. Im Datenmodell ist jedoch vorgesehen, dass zu jeder Kamera ein Netzteil passt und auch angegeben wird. Er stellt also fest, dass die Klassifizierung so noch nicht optimal ist.

Um die Klassifizierung den neu gewonnenen Erkenntnissen anzupassen, muss er die Modellierung der Kameras verfeinern: Es braucht zur Kamera jeweils eine Subklasse für Solarbetriebene Kameras und eine für Konventionelle. Ausserdem müssen die Attribute passend auf die Klassen verteilt werden und ein neues Attribut (Leistung der Solarzellen) modelliert werden.

Kevin erstellt die neuen Klassen und lässt diese von der bestehenden Kamera Klasse erben. Ausserdem hängt er alle bestehenden Produkte an die Konventionelle Kamera Klasse. Für das neue Produkt erstellt er eine Klasse und lässt diese von der Solarkamera erben.

Als nächstes verschiebt Kevin die Attribute welche das Netzteil beschreiben auf die Konventionelle Kamera.

Nun möchte Kevin noch das Attribut für die Leistung der Solarzellen erfassen. Um sicherzustellen, dass es nicht mehrere redundante Attributdefinitionen gibt, durchsucht Kevin die 2485 bestehenden Attribute und prüft, ob in einer anderen Produktkategorie bereits ein passendes Attribut verwendet wurde. Er wird fündig und fügt das Attribut „Solarzellenleistung“ der Solarkamera Klasse hinzu.

Zufrieden mit der verfeinerten Datenmodellierung widmet sich Kevin wieder dem neuen Verkaufskatalog von Dagitec.

Scenario 4

Masken

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Dagitec verkauft Computerzubehör und als Spezial Nische Raritäten aus der Autowelt über einen Webshop. Die Produktstrukturierung ist schon recht weit fortgeschritten nächste Woche findet das vierte Meeting mit Dagitec statt.

Kevin ist mit L.K. einem Logistiker in einem Meeting. Gemeinsam wollen sie den Produkttyp Kompaktkamera besprechen. Da L.K. kein UML versteht, möchte Kevin eine Eingabemaske zum Erfassen solcher

Kompaktkameras generieren.

Kevin generiert auf Knopfdruck eine solche Maske, speziell für den Logistiker. L.K. überprüft diese Maske und stellt fest, dass ein Attribut fehlt.

Kevin wechselt zur Attributvererbungsansicht und findet innert kürzester Zeit die Klasse, welche dieses Attribut enthält. Er stellt auch sofort fest, dass der Produkttyp Kompaktkamera von dieser Klasse erbt. Als nächstes überprüft er die Metaattribute. Er stellt fest, dass im Metaattribut „Read / Write – Rechte“ die Rolle des Logistikers nicht erfasst ist. Kevin bereinigt dies und L.K. macht ihn darauf aufmerksam, dass das Attribut nur einmalig gesetzt werden darf. Kevin setzt das „Write-Once“ – Metaattribut und generiert die neue Eingabemaske.

2.2 Umfang

Hier beschreiben wir welche Features auf welchen Tabs vorhanden sind.

- Modellieren von Attributen
  - Attribute verwalten
  - Metaattributen zu Attributen verwalten
  - Attribute suchen
- Modellieren einer Attributvererbungshierarchie
  - Modellieren von UML-Klassen mit Vererbung
  - Attribute der Klassen verwalten
  - Metaattribute der Klassen verwalten
- Modellieren eines Produkttyps
  - Modellieren von Produkttypen mit allen Attributen durch Vererbung
  - Metaattribute der Produkttypen verwalten
  - Anzeigen des Produkttyps für nicht Informatiker (Masken)

2.3 Navigationsstruktur

Da der Kunde noch nicht genau weiss, welche Features er noch benötigen wird, haben wir eine einfache Navigationsstruktur, ohne Querverweise und durchdachte Usability, entwickelt.

Die oberste Ebene bildet eine Reihe von Tabs. Die Punkte darunter sind jeweils Subansichten.

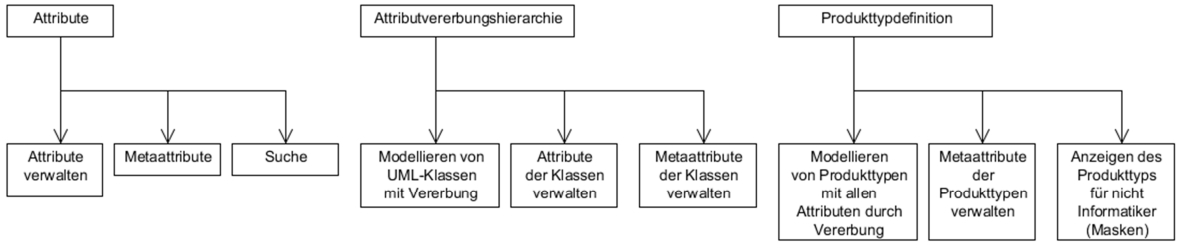


Abbildung 1 Navigationsstruktur nach Garrett

2.4 Raster

Alle unsere Screens sind anhand des folgenden Rasters aufgebaut. Dieses Raster unterstützt den User sich zu orientieren.

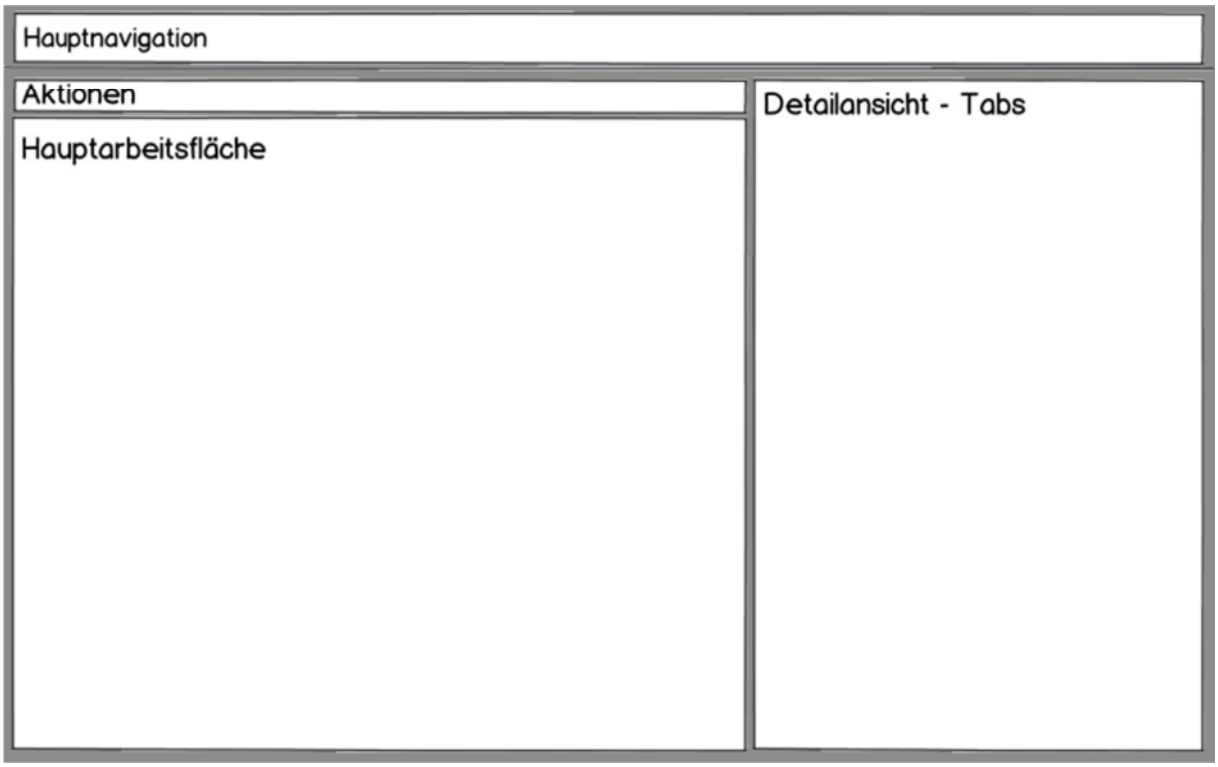


Abbildung 2 Raster nach Garrett

2.5 Oberfläche (Wireframe Prototype)

In diesem Projekt haben wir nur Wireframes entworfen. Weitere Themen wie z.B. ein Farbschema oder Schriften haben wir ignoriert. Die Interessen des Users fokussieren sich auf die Funktionalitäten dieses Programms. Für ihn ist es wichtig zu sehen wie man so eine Modellierung durchführen und wie die Daten darstellen könnte. Die Usability steht nicht im Vordergrund. Wir dürfen davon ausgehen, dass sich der User mit dem Thema Produktstrukturierung auskennt und sich mit dem Programm vertraut macht.

In den folgenden Wireframes wurden die Suchmaske sowie die Eingabemaske, welche automatisch generiert wird nicht dargestellt, da diese nur die Idee der Tabs demonstrieren sollen.

2.5.1 Attribute

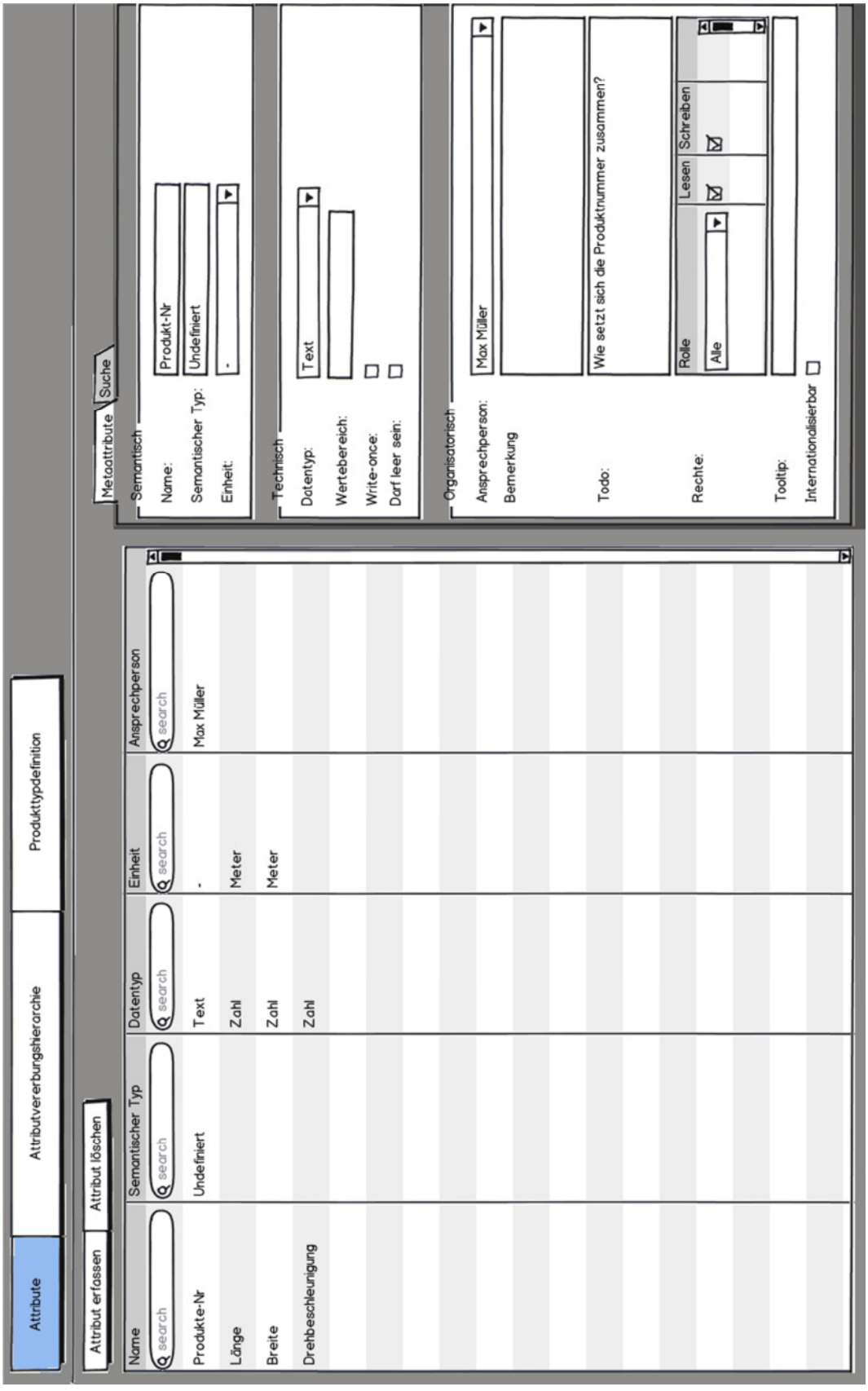


Abbildung 3 Attribute Ansicht Version 4

2.5.2 Attributvererbungshierarchie

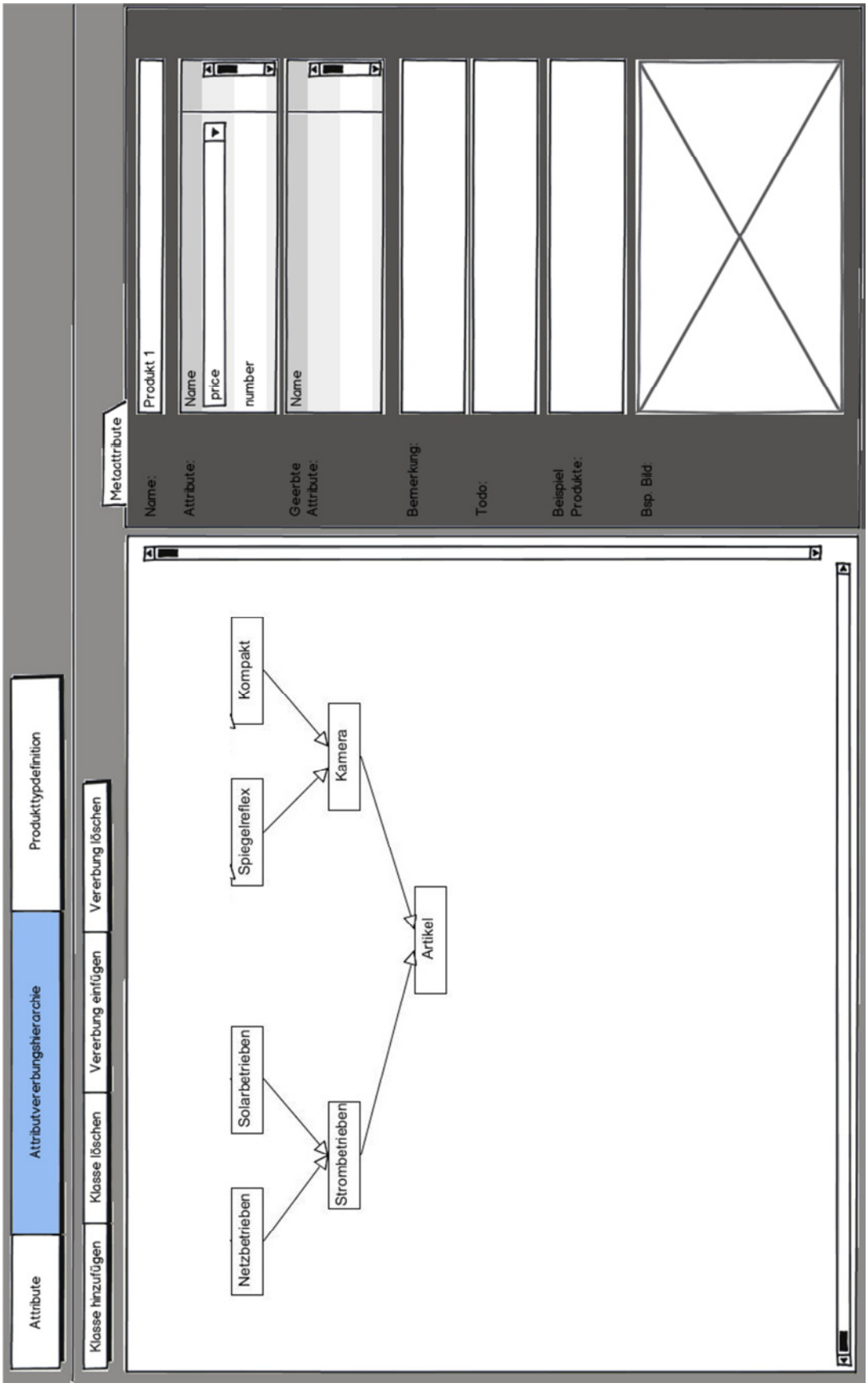


Abbildung 4 Attributvererbungshierarchie Version 4

2.5.3 Produkttypdefinition

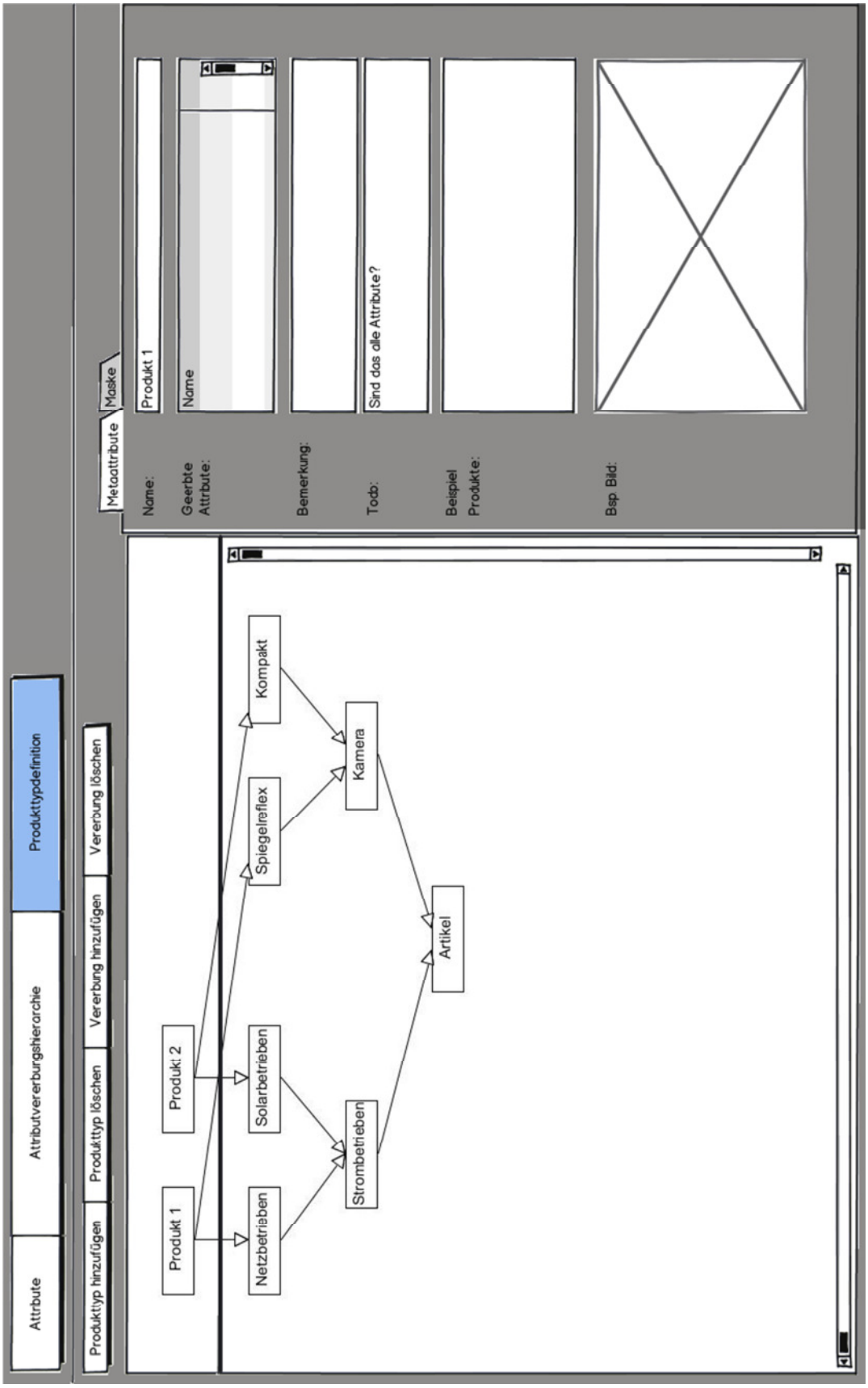


Abbildung 5 Produkttypdefinition Version 4

3 Entwicklung Ist-Zustand, Benutzer, Soll-Szenarien

3.1 Erste Iteration

3.1.1 Heutige Arbeitsweise

Heutige Arbeitsweise

Eine Produktstrukturierung findet heute in etwa nach folgenden Schritten statt:

- Es werden Ziele und Einschränkungen festgelegt (welche Systeme sollen neu gemacht werden, welche müssen bestehen bleiben)
- Sitzungen für gemeinsames Verständnis von Begriffen (wobei die nicht sauber und exakt definiert werden)
- Der Kunde erstellt eine Liste von Attributen (ohne weitere Definitionen) in Excel.
- fyayc konfiguriert PIM

Als Grundlage für diese Arbeiten dienen Kataloge, der Webshop und Umsysteme.

Probleme

- Die Arbeiten finden unstrukturiert statt. Weder gibt es eine einheitliche Vorgehensweise, noch werden die Resultate sauber strukturiert dokumentiert.
- Verschiedene Begriffe sind nicht sauber definiert und werden für verschiedene Konzepte eingesetzt. Im Gegenzug gibt es für einige Konzepte auch keine Begriffe.

3.1.2 Benutzer

Benutzer

- Er hat ein überdurchschnittliches logisch-abstraktes Denkvermögen.
- Er weiss wie man eine Produktstrukturierung durchführt und wozu.
- Er hat eine (kurze) Einführung in PSA erhalten und kennt die verwendeten Begriffe.

3.1.3 Szenarios

Szenario 1: Neues Attribut erfassen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist schon recht weit fortgeschritten, nächste Woche findet das vierte Meeting mit Dagitec statt.

Kevin war letzte Woche im Ausland und beauftragte seinen Assistenten, das aktuelle PIM-Datenmodell mit dem Webshop von Dagitec abzugleichen um eventuell vergessene Attribute zu finden. Als er am Montagmorgen frisch an die Arbeit möchte, liegt eine Notiz von seines Assistenten auf seinem Schreibtisch in welcher steht, dass seiner Meinung nach das Farb-Attribut vergessen wurde, denn heutzutage könne man den Ford T auch in Pink kaufen.

Kevin startet sofort die PSA-Applikation und überprüft ob dieses

Attribut wirklich nicht existiert. Dazu öffnet er das Suchfenster und testet mehrere mögliche Synonyme. Tatsächlich scheint dieses Attribut nicht zu existieren. Kevin erfasst das neue Attribut im System und erfasst auch gleich die Metaattribute:

- Attributtyp: String
- Muss übersetzt werden: ja

Damit die Farbe Pink nicht ausversehen für ein anderes Fahrzeug verwendet wird, erfasst Kevin noch ein weiteres Metaattribut "Hinweis".

Zufrieden fügt er das neue Attribut in der Klasse Auto ein.

http://de.wikipedia.org/wiki/Ford_Modell_T

Szenario 2: Attribut Suchen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist schon recht weit fortgeschritten nächste Woche findet das fünfte Meeting mit Dagitec statt.

Kevin erfasst mit dem Kunden gerade eine neue Klasse. Beim Einfügen des neuen Attributes Länge haben die Beiden verschiedenen Attributnamen für dasselbe Attribut im Kopf. Kevin möchte es "Innenlänge" nennen, da es ihm der Kunde auch so erklärt hat. Der Kunde findet aber "Länge" passender, denn niemand würde in der Praxis von "Innenlänge" sprechen. Um zu überprüfen, ob es schon ein solches Attribut mit der gewünschten Bedeutung gibt, wechseln sie auf die Suchmaske und suchen nach den beiden Namen. Tatsächlich existieren beide Attribute. Um deren Bedeutung genauer zu verstehen, schauen sie sich die erfassten Metaattribute an. Die beiden Attribute scheinen auf den ersten Blick sehr ähnlich und sie werden sich nicht einig, welches Attribut sie jetzt verwenden sollen. Darauf ruft Kevin beim Erfasser dieser beiden Attribute an und fragt nach deren genauen Bedeutung. Jetzt weiss er, dass er das Attribut "Innenlänge" verwenden muss. Er notiert seine Erkenntnisse sofort im Kommentarfeld des Attributes.

Szenario 3: Neue Klasse erstellen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist schon recht weit fortgeschritten nächste Woche findet das dritte Meeting mit Dagitec statt.

Rechtzeitig zur Osterzeit bringt Dagitec einen neuen Katalog mit top aktuellen Produkten auf den Markt. K.H. von der Firma Dagitec schickt die neue Auflage an Kevin. Am nächsten Tag blättert Kevin den neuen Katalog durch. Darin findet er ein neues Produkt: Eine solarbetriebene Fotokamera. Zunächst begeistert von dieser Innovation, bemerkt er anschliessend aber, dass man zu diesem Gerät kein Netzteil kaufen kann. Im Datenmodell ist jedoch vorgesehen,

dass zu jeder Kamera ein Netzteil passt und auch angegeben wird. Er stellt also fest, dass die Klassifizierung so noch nicht optimal ist.

Um die Klassifizierung den neu gewonnenen Erkenntnissen anzupassen, muss er die Modellierung der Kameras verfeinern: Es braucht zur Kamera jeweils eine Subklasse für Solarbetriebene Kameras und eine für Konventionelle. Ausserdem müssen die Attribute passend auf die Klassen verteilt werden und ein neues Attribut (Leistung der Solarzellen) modelliert werden.

Kevin erstellt die neuen Klassen und lässt diese von der bestehenden Kamera Klasse erben. Ausserdem hängt er alle bestehenden Produkte an die Konventionelle Kamera Klasse. Für das neue Produkt erstellt er eine Klasse und lässt diese von der Solarkamera erben.

Als nächstes verschiebt Kevin die Attribute welche das Netzteil beschreiben auf die Konventionelle Kamera.

Nun möchte Kevin noch das Attribut für die Leistung der Solarzellen erfassen. Um sicherzustellen, dass es nicht mehrere redundante Attributdefinitionen gibt, durchsucht Kevin die 2485 bestehenden Attribute und prüft, ob in einer anderen Produktkategorie bereits ein passendes Attribut verwendet wurde. Er wird fündig und fügt das Attribut der Solarkamera Klasse hinzu.

Zufrieden mit der verfeinerten Datenmodellierung widmet sich Kevin wieder dem neuen Verkaufskatalog von Dagitec.

3.1.4 Wireframes

Um mit dem User konkreter diskutieren zu können, haben wir erste Eingabemaske und Anzeigemöglichkeiten als Wireframes modelliert.

Mit diesen Wireframes wird die Hauptfunktionalität in allen Szenarien abgedeckt.

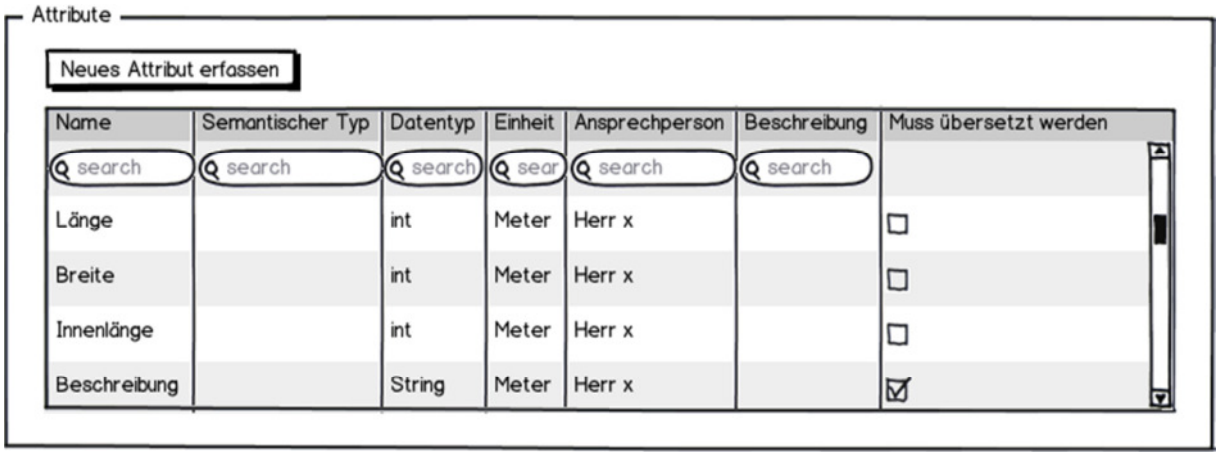


Abbildung 6: Attribut Suchen

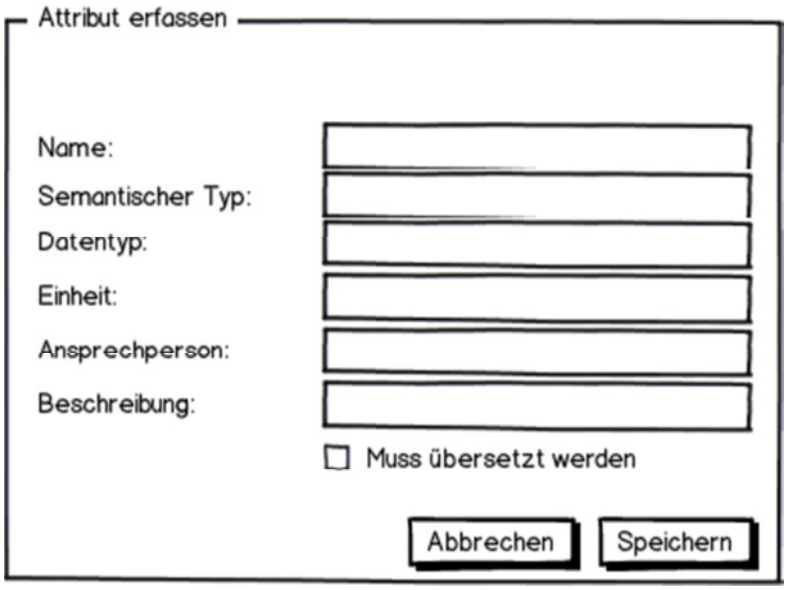


Abbildung 7: Attribut erfassen

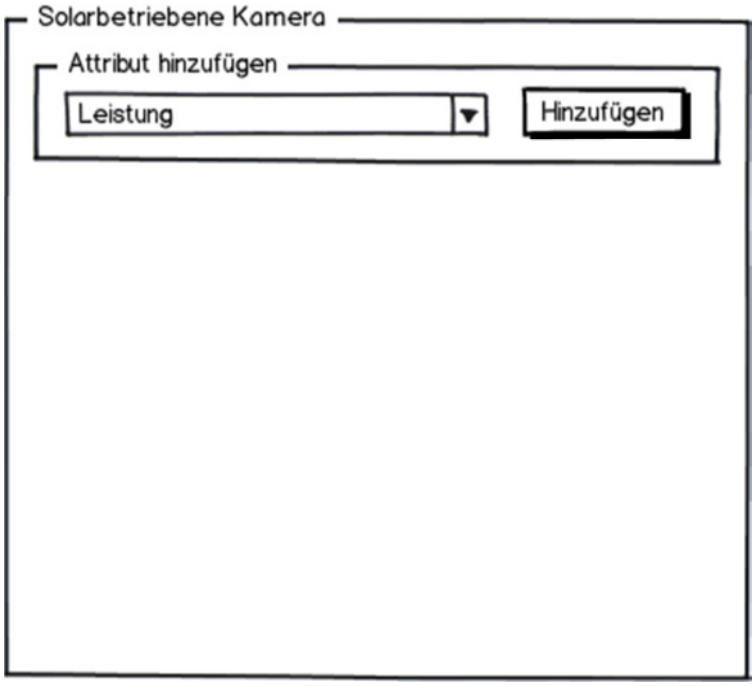


Abbildung 8: Attribut zu Klasse hinzufügen

Um das Szenario 3 zu unterstützen haben wir hier unsre Vorstellung für die Modellierung der Kameras gezeichnet.

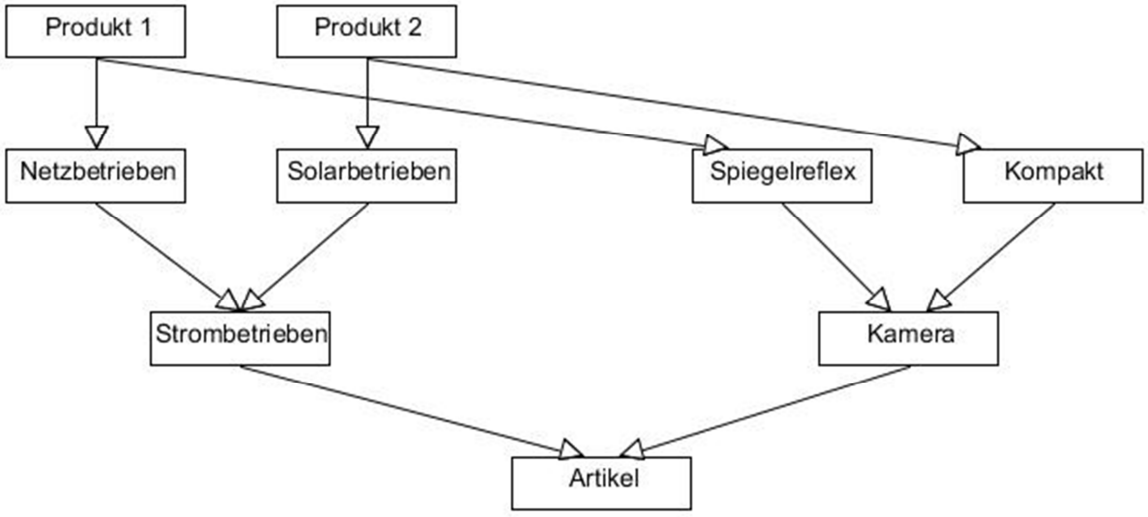


Abbildung 9: Attributvererbungshierarchie

3.1.5 Diskussion mit D. Keller

3.1.5.1 Vorbereitete Fragen

Frage	Antwort
Wie / Unter welchen Umständen soll PSA eingesetzt werden? • Von fyayc Mitarbeiter alleine? (Nachbearbeitung von Meetings, Modellieren seiner Erkenntnisse) • Oder: Von fyayc-Mitarbeiter während Meeting, d.h. die Mitarbeiter des Kunden sehen zu? • Oder: Von Kunden-Mitarbeiter alleine (Es ist seine Aufgabe die Attribute zu erfassen)?	Das Produkt soll von einem fyayc Mitarbeiter bedient werden.

Wozu erfasst und beschreibt man die Attribute? Wozu erstellt man eine Attributvererbungshierarchie?
Unsere Vorstellung ist:
• Finden aller Attribute
• Konsistente Namensgebung der Attribute
• Verständnis der Produktstruktur verbessern. Strukturierung des Mindmodel.
• Vorarbeit für Erstellung des logischen Domainmodells
  ○ Attribute
  ○ Datentypen
  ○ wie Vererbung Umsetzen
  ○ ... was fehlt hier noch?
• Hilfestellung für Erstellung der Navigationshierarchie
• Hilfestellung für Erstellung Faceted Search
• Begriffsdefinitionen
• Zusammenhang zwischen den Produkten aufzeigen
  ○ Zubehör
  ○ Bestandteil (Composite)
  ○ Variation
  ○ alternativ Produkte
  ○ ... was fehlt hier noch?
	Genau. Ausserdem:
	• Hilfestellung für Auswertungen (z.B. kann man im laufendem Betrieb Absatzstatistiken erstellen)
	• Vereinfachtes Hinzufügen neuer Produkte (im laufenden Betrieb) (Man muss nicht mehr alle nötigen Attribute angeben sondern muss nur noch sagen, von welchen Klassen geerbt werden soll.)
	• Die Definition von Eingabemasken wird unterstützt oder sogar automatisiert (Z.B. kann anhand der Metaattribute eine Eingabemaske für einen spezifischen Mitarbeiter generiert werden, welchem nur diejenigen Attribute angezeigt werden, welche für ihn auch relevant sind).

Welchen Einfluss haben die Beschreibung der Attribute und die Attributvererbungshierarchie auf das PIM und dessen OR-Mapper?	

3.1.5.2 Weitere Diskussionspunkte

- Heutige Arbeitsweise
- Einsatzkontext von PSA (Einsatz während Projektablauf, Beschreibung der Benutzer (Hauptsächlich fyayc-Mitarbeiter, neu für gewisse Aufgaben auch der Kundenmitarbeiter)
- Validierung der Szenarien (D. Keller findet die Szenarien gut, es sind nur kleine Anpassungen nötig)
- Input für ein neues Szenario: Diskussion der Modellierung eines Produkttyps mit einem Kundenmitarbeiter anhand einer Eingabemaske (Formulare)

3.2 Zweite Iteration

Die Workproducts der Iteration 1 haben wir anhand den Inputs von D. Keller verbessert und hier die überarbeiteten Versionen festgehalten.

3.2.1 Heutige Arbeitsweise

Heutige Arbeitsweise

Eine Produktstrukturierung findet heute in etwa nach folgenden Schritten statt:

- Die heutige IT-Landkarte wird aufgezeigt und bestimmt welche Systeme beibehalten, welche verändert und welche neu gemacht werden sollen
- Es werden Ziele festgelegt z.B.:
  - Bessere Usability
  - Weniger Retouren
  - Schnellere Lieferungen
  - Bessere Kundenbindung
- Es werden Einschränkungen festgelegt z.B.:
  - Daten die unter Datenschutz stehen
  - Technologieeinschränkungen
- Requirements und UseCases
- Evaluation des Implementationspartner
- Es werden Ziele und Einschränkungen festgelegt (welche Systeme sollen neu gemacht werden, welche müssen bestehen bleiben)
- Sitzungen für gemeinsames Verständnis von Begriffen (wobei die nicht sauber und exakt definiert werden)
- Der Kunde erstellt eine Liste von Attributen (ohne weitere Definitionen) in Excel. (→ Hier wird PSA ansetzen)
- Vorgehen wird besprochen
  - Iterationen planen (1-2 Wochen für fyayc Entwickler)
  - Meilensteine für Kunden alle 3 Monate ein Produktives Release z.B. nur in einzelnen Ländern
- In Iterationen werden die Systeme von fyayc konfiguriert: PIM, CRM, MAM, ERP, Webshop, Supportsystem, etc.

Als Grundlage für diese Arbeiten dienen Kataloge, der Webshop und Umsysteme.

Probleme

- Die Arbeiten finden unstrukturiert statt. Weder gibt es eine einheitliche Vorgehensweise, noch werden die Resultate sauber strukturiert dokumentiert.
- Verschiedene Begriffe sind nicht sauber definiert und werden für verschiedene Konzepte eingesetzt. Im Gegenzug gibt es für einige Konzepte auch keine Begriffe.

3.2.2 Einsatzkontext von PSA

Einsatz während Projektablauf

Im obigen Ablauf (Heutige Arbeitsweise) findet PSA ab der Erstellung der Attributliste Verwendung.

Benutzer

Für die Modellierung wird PSA von fyayc Mitarbeitern verwendet. Es kann nützlich sein, gewisse Visualisierungen mit dem Kunden zu besprechen.

fyayc-Benutzer

- Er hat ein überdurchschnittliches logisch-abstraktes Denkvermögen.
- Er weiss wie man eine Produktstrukturierung durchführt und wozu.
- Er hat eine (kurze) Einführung in PSA erhalten und kennt die verwendeten Begriffe.

Kundenmitarbeiter

Bei Darstellungen welche für Diskussionen mit dem Kunden gedacht sind, darf von keinen spezifischen Kenntnissen ausgegangen werden.

Ziel

Folgende Ziele sollen durch den Einsatz von PSA einfacher erreicht werden:

- Begriffsdefinitionen finden (zusammen mit Kunden)
- Finden aller Attribute
- Konsistente Namensgebung der Attribute
- Zusammenhang zwischen den Produkten aufzeigen
  - Zubehör
  - Bestandteil (Composite)
  - Variation
  - alternativ Produkte
- Verständnis der Produktstruktur verbessern. Strukturierung des Mindmodel.
- Vorarbeit für Erstellung des logischen Domainmodells
  - Attribute
  - Datentypen
  - wie Vererbung Umsetzen
- Hilfestellung für Erstellung der Navigationshierarchie
- Hilfestellung für Erstellung Faceted Search
- Hilfestellung für Auswertungen
  - Z.B. kann man im laufenden Betrieb Absatzstatistiken erstellen
- Vereinfachtes Hinzufügen neuer Produkte (im laufenden Betrieb) (Man muss nicht mehr alle nötigen Attribute angeben sondern muss nur noch sagen, von welchen Klassen geerbt werden soll.)
- Die Definition von Eingabemasken wird unterstützt oder sogar automatisiert (Z.B. kann anhand der Metaattribute eine Eingabemaske für einen spezifischen Mitarbeiter generiert werden, welchem nur diejenigen Attribute angezeigt werden, welche für ihn auch relevant sind).

3.2.3 Szenarios

Diese Szenarien haben wir anhand den Inputs in Iteration eins von D. Keller überarbeitet und erneut validiert.

Szenario 1: Neues Attribut erfassen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist schon recht weit fortgeschritten, nächste Woche findet das vierte Meeting mit Dagitec statt.

Kevin liest in der Zeitung, dass der Ford T jetzt auch in Pink erhältlich sei. Da er sich nicht sicher ist, ob das aktuelle Datenmodell diese Konfiguration zulässt, startet er die PSA-Applikation und überprüft ob es ein Attribut „Farbe“ für dieses Modell gibt. Dazu öffnet er das Suchfenster und testet mehrere mögliche Synonyme. Tatsächlich scheint kein exakt passendes Attribut zu existieren. Kevin erfasst das neue Attribut im System und erfasst auch gleich die Metaattribute:

- Attributtyp: String
- Muss übersetzt werden: ja

Damit die Farbe Pink nicht ausversehen für ein anderes Fahrzeug verwendet wird, erfasst Kevin noch ein weiteres Metaattribut “Hinweis”.

Zufrieden fügt er das neue Attribut in der Klasse Auto ein.

http://de.wikipedia.org/wiki/Ford_Modell_T

Szenario 2: Attribut Suchen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist noch nicht weit fortgeschritten nächste Woche findet das zweite Meeting mit Dagitec statt.

Kevin studiert mit einem Kunden dessen Produktkatalog und erfasst mit ihm eine neue Klasse. Beim Einfügen des neuen Attributes Länge haben die Beiden verschiedenen Attributnamen für dasselbe Attribut im Kopf. Kevin möchte es “Innenlänge” nennen, da es ihm der Kunde auch so erklärt hat. Der Kunde findet aber “Länge” passender, denn niemand würde in der Praxis von “Innenlänge” sprechen. Um zu überprüfen, ob es schon ein solches Attribut mit der gewünschten Bedeutung gibt, wechseln sie auf die Suchmaske und suchen nach den beiden Namen. Tatsächlich existieren beide Attribute. Um deren Bedeutung genauer zu verstehen, schauen sie sich die erfassten Metaattribute an. Die beiden Attribute scheinen auf den ersten Blick sehr ähnlich und sie werden sich nicht einig, welches Attribut sie jetzt verwenden sollen. Darauf ruft Kevin beim Erfasser dieser beiden Attribute an und fragt nach deren genauen Bedeutung. Jetzt weiss er, dass er das Attribut “Innenlänge” verwenden muss. Er notiert seine Erkenntnisse sofort im Kommentarfeld des Attributes.

Szenario 3: Neue Klasse erstellen

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist schon recht weit fortgeschritten nächste Woche findet das dritte Meeting mit Dagitec statt.

Rechtzeitig zur Osterzeit bringt Dagitec einen neuen Katalog mit top aktuellen Produkten auf den Markt. K.H. von der Firma Dagitec schickt die neue Auflage an Kevin. Am nächsten Tag blättert Kevin den neuen Katalog durch. Darin findet er ein neues Produkt: Eine solarbetriebene Fotokamera. Zunächst begeistert von dieser Innovation, bemerkt er anschliessend aber, dass man zu diesem Gerät kein Netzteil kaufen kann. Im Datenmodell ist jedoch vorgesehen, dass zu jeder Kamera ein Netzteil passt und auch angegeben wird. Er stellt also fest, dass die Klassifizierung so noch nicht optimal ist.

Um die Klassifizierung den neu gewonnenen Erkenntnissen anzupassen, muss er die Modellierung der Kameras verfeinern: Es braucht zur Kamera jeweils eine Subklasse für Solarbetriebene Kameras und eine für Konventionelle. Ausserdem müssen die Attribute passend auf die Klassen verteilt werden und ein neues Attribut (Leistung der Solarzellen) modelliert werden.

Kevin erstellt die neuen Klassen und lässt diese von der bestehenden Kamera Klasse erben. Ausserdem hängt er alle bestehenden Produkte an die Konventionelle Kamera Klasse. Für das neue Produkt erstellt er eine Klasse und lässt diese von der Solarkamera erben.

Als nächstes verschiebt Kevin die Attribute welche das Netzteil beschreiben auf die Konventionelle Kamera.

Nun möchte Kevin noch das Attribut für die Leistung der Solarzellen erfassen. Um sicherzustellen, dass es nicht mehrere redundante Attributdefinitionen gibt, durchsucht Kevin die 2485 bestehenden Attribute und prüft, ob in einer anderen Produktkategorie bereits ein passendes Attribut verwendet wurde. Er wird fündig und fügt das Attribut der Solarkamera Klasse hinzu.

Zufrieden mit der verfeinerten Datenmodellierung widmet sich Kevin wieder dem neuen Verkaufskatalog von Dagitec.

Scenario 4

Masken

Die Firma StruVor entwickelt zurzeit für die Firma Dagitec ein neues PIM-Datenmodell. Die Produktstrukturierung ist schon recht weit fortgeschritten nächste Woche findet das vierte Meeting mit Dagitec statt.

Kevin ist mit L.K. einem Logistiker in einem Meeting. Gemeinsam wollen sie den Produkttyp Kompaktkamera besprechen. Da L.K. kein UML versteht, möchte Kevin eine Eingabemaske zum Erfassen solcher Kompaktkameras generieren.

Kevin generiert auf Knopfdruck eine solche Maske, speziell für den

Logistiker. L.K. überprüft diese Maske und stellt fest, dass ein Attribut fehlt.

Kevin wechselt zur Attributvererbungsansicht und findet innert kürzester Zeit die Klasse, welche dieses Attribut enthält. Er stellt auch sofort fest, dass der Produkttyp Kompaktkamera von dieser Klasse erbt. Als nächstes überprüft er die Metaattribute. Er stellt fest, dass im Metaattribut „Read / Write – Rechte“ die Rolle des Logistikers nicht erfasst ist. Kevin bereinigt dies und L.K. macht ihn darauf aufmerksam, dass das Attribut nur einmalig gesetzt werden darf. Kevin setzt das „Write-Once“ – Metaattribut und generiert die neue Eingabemaske.

3.2.4 Diskussion mit D. Keller

3.2.4.1 Vorbereitete Fragen

Frage	Antwort
Was heisst ihr Arbeitet unstrukturiert? Was empfinden sie als unstrukturiert?	Man ist wenig organisiert, hat kein Standard-Prozess
Wir sehen ein Problem darin, wenn jemand für 2500 Attribute alle Metaattribute erfassen muss. Können wir dieses Problem auf eine nächste Iteration verschieben?	Ja

3.2.4.2 Weitere Diskussionspunkte

- Arbeitsmaterialen
- D. Keller findet die Szenarien wie sie sind gut und hat keinen weiteren Input gegeben

3.3 Dritte Iteration

Die Diskussion hat ergeben, dass die bis anhin erarbeiteten Resultate gut sind. Die wenigen neuen Inputs haben wir noch einfliessen lassen und im Kapitel „2 Resultat“ festgehalten.

4 Entwicklung Prototyp

4.1 Variante 1

In dieser Ansicht kann der Benutzer Metaattribute zu Attributtypen erfassen. Sie diente uns als Diskussionsgrundlage für die dritte Sitzung mit Herrn Keller.

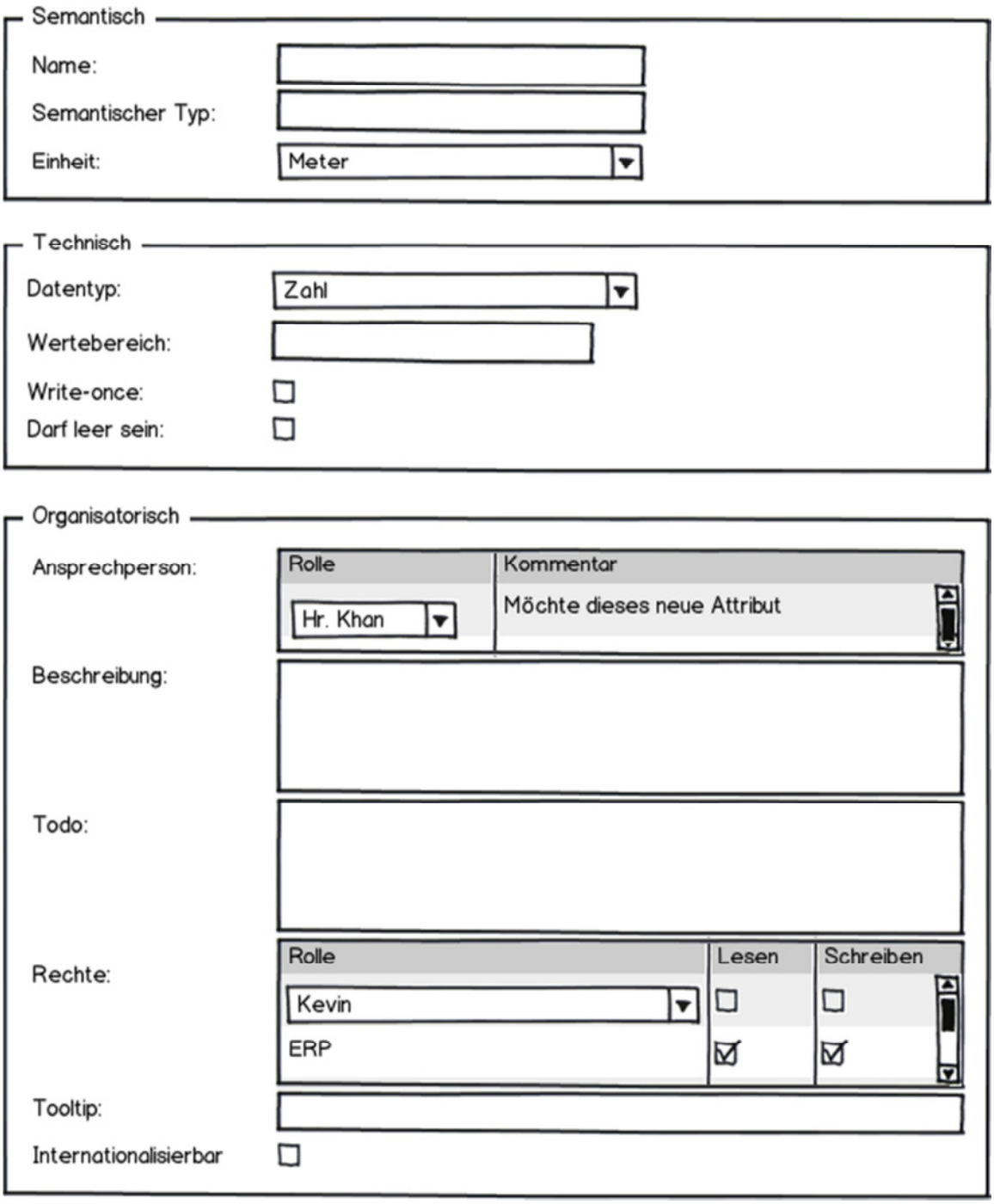


Abbildung 10 Metaattribute Variante 1

4.2 Variante 2 (Vorbereitung auf den heuristischen Test)

Dies ist unser erster vollständiger Prototyp. Michael Steiner hat den Prototyp entworfen. Christoph Rosenberger hat den heuristischen Test (cognitive Walktrough) durchgeführt.

4.2.1 Attributdetail Ansicht

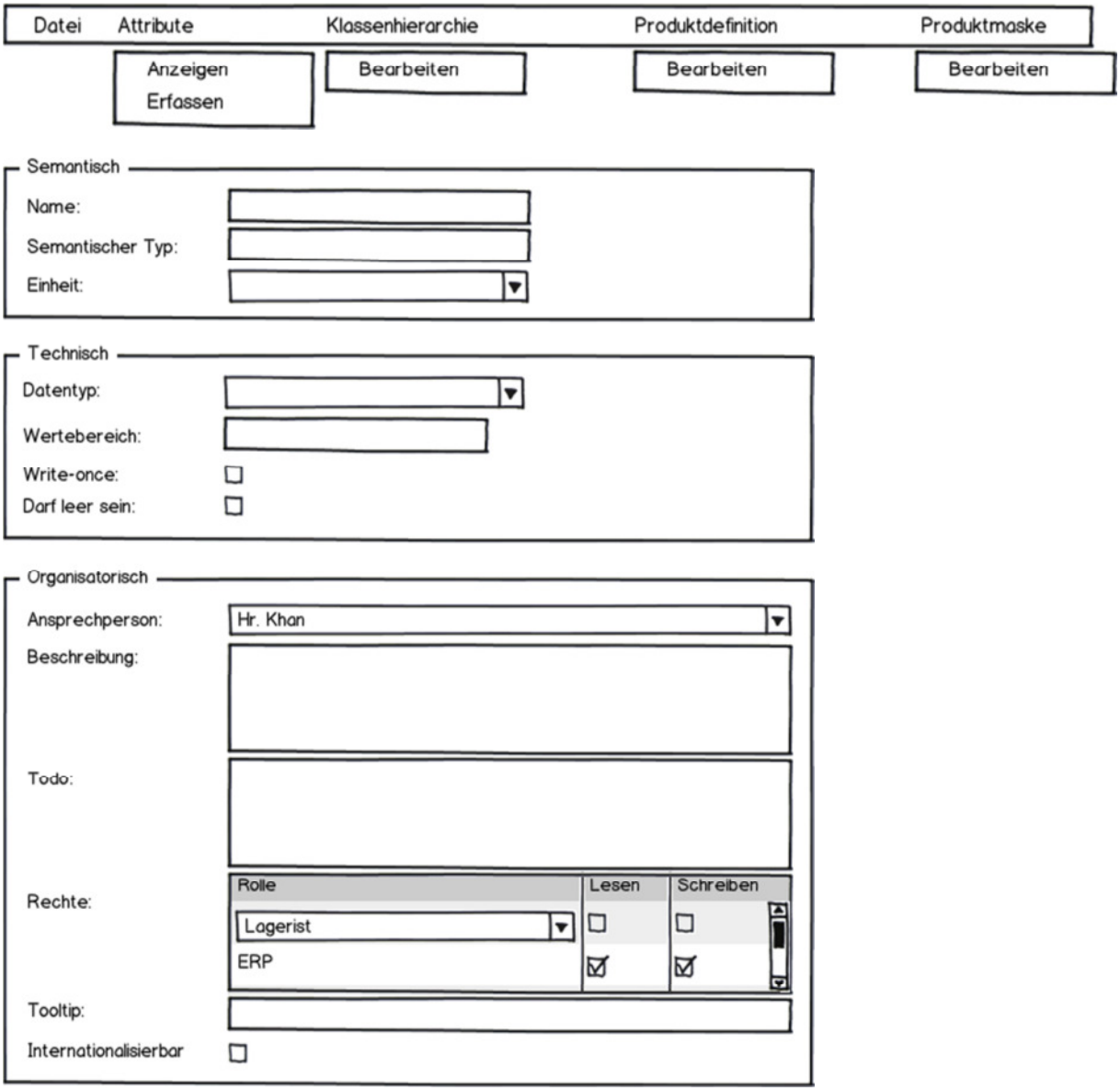


Abbildung 11 Metaattribute Variante 2

4.2.2 Attributliste

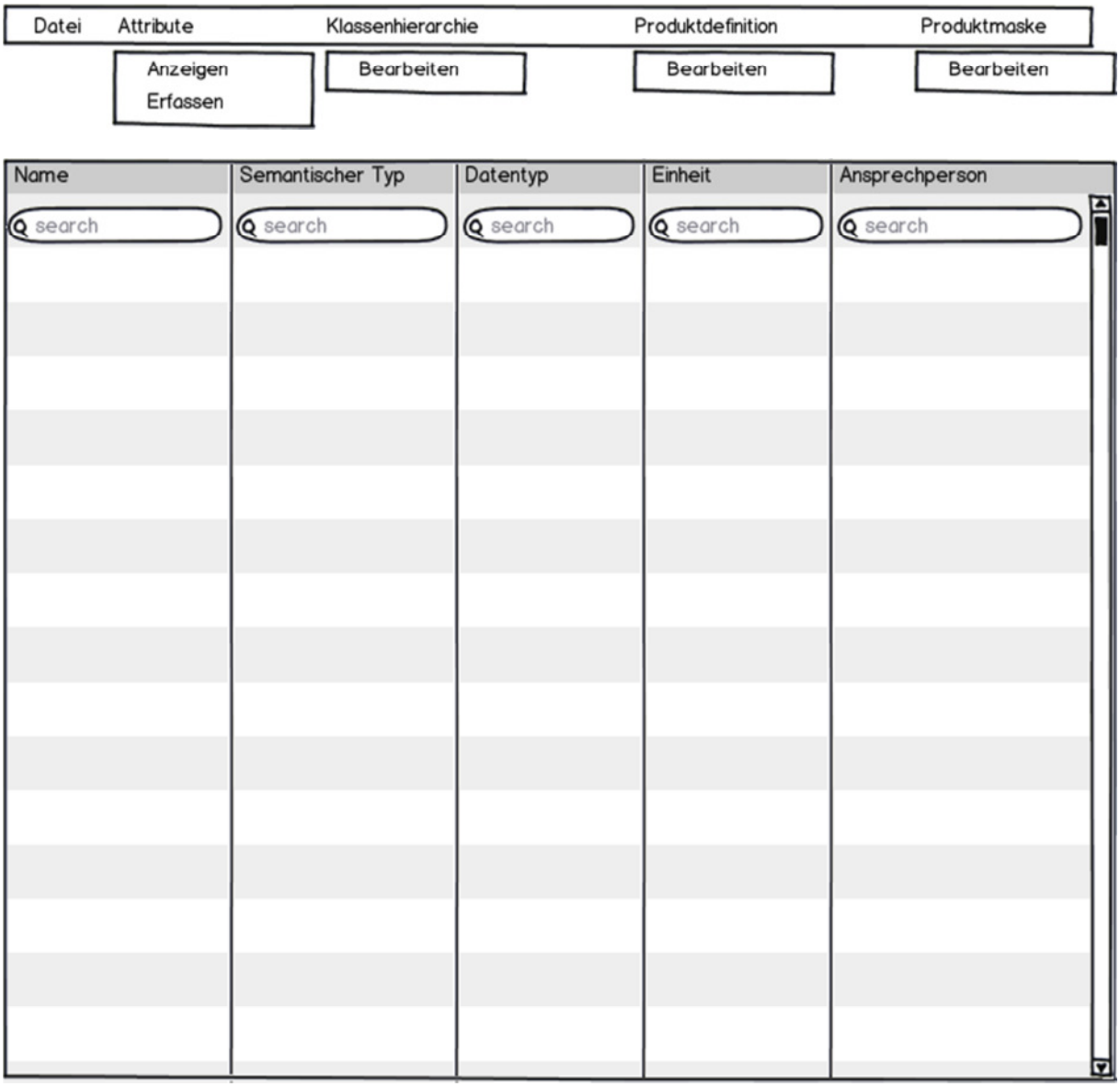


Abbildung 12 Attributliste Variante 2

4.2.3 Attributvererbungshierarchie

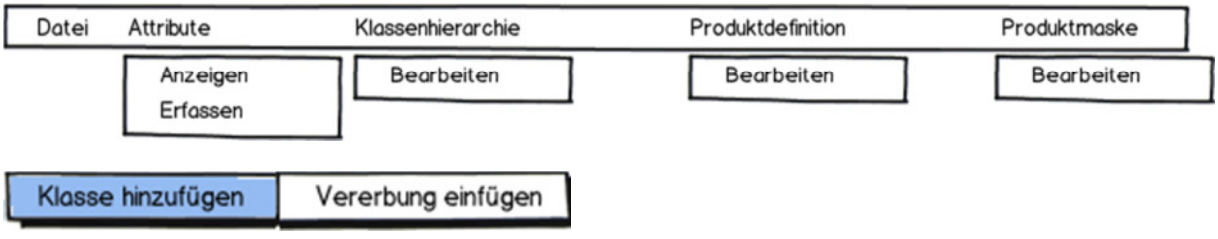


Abbildung 13 Attributvererbungshierarchie Variante 2

4.2.4 Attributvererbungshierarchiedetail

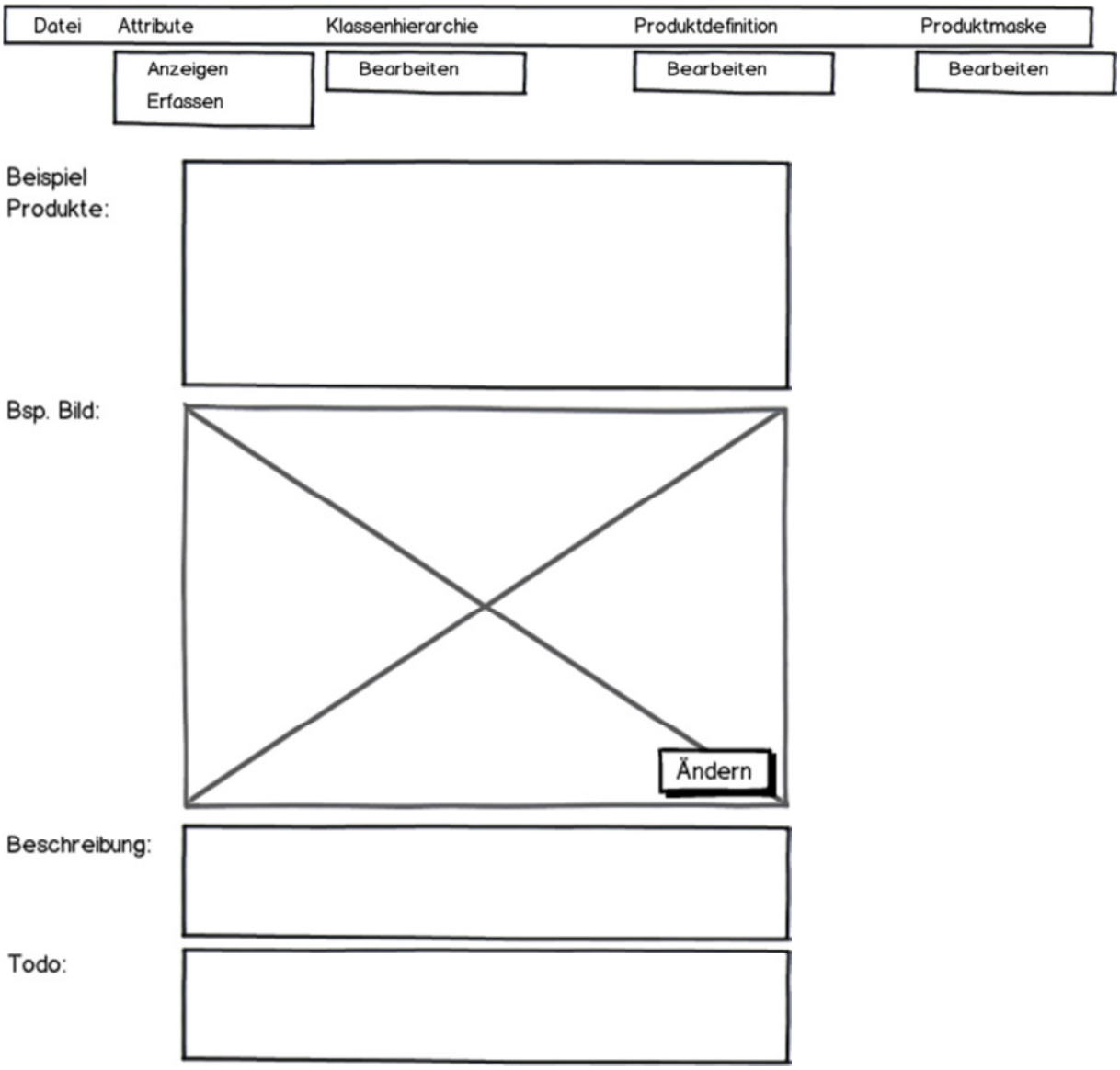


Abbildung 14 Attributvererbungshierarchiedetail Variante 2

4.2.5 Diskussion

- Die Navigation mit den Dropdowns ist so nicht gut
- Zum Teil fehlen wichtige Bedienelemente z.B. Attribute erfassen und löschen
- Mehr Metaattribute
- Screen für die Maskenansicht und die Produkttypdefinitionen fehlen

4.3 Variante 3 (Test mit Hr. Keller)

Diese Variante haben wir mit Herrn Keller in einem Papierprototyptest durchgespielt.

4.3.1 Attributdetail Ansicht

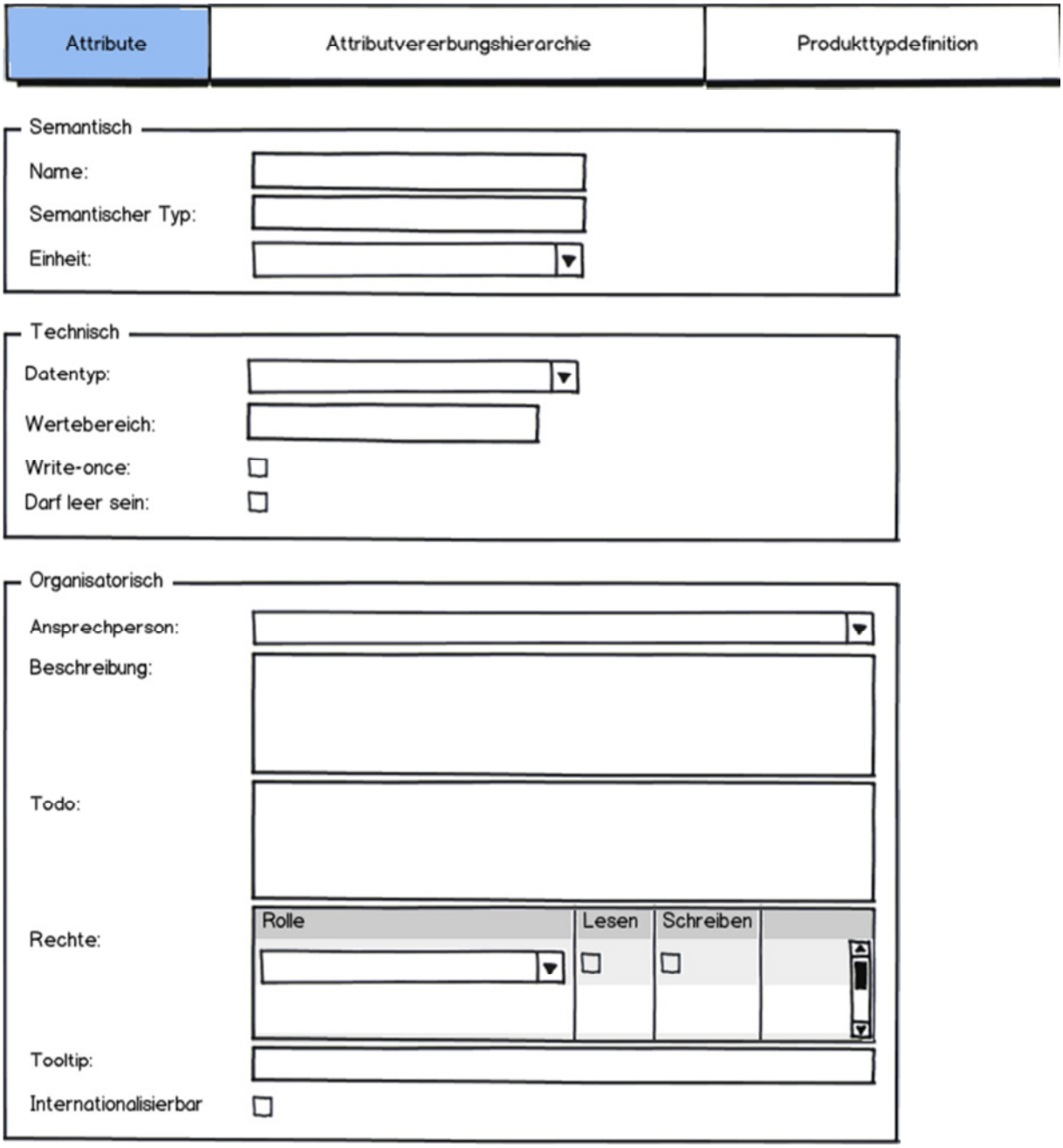


Abbildung 15 Attributdetailansicht Variante 3

4.3.2 Attributliste

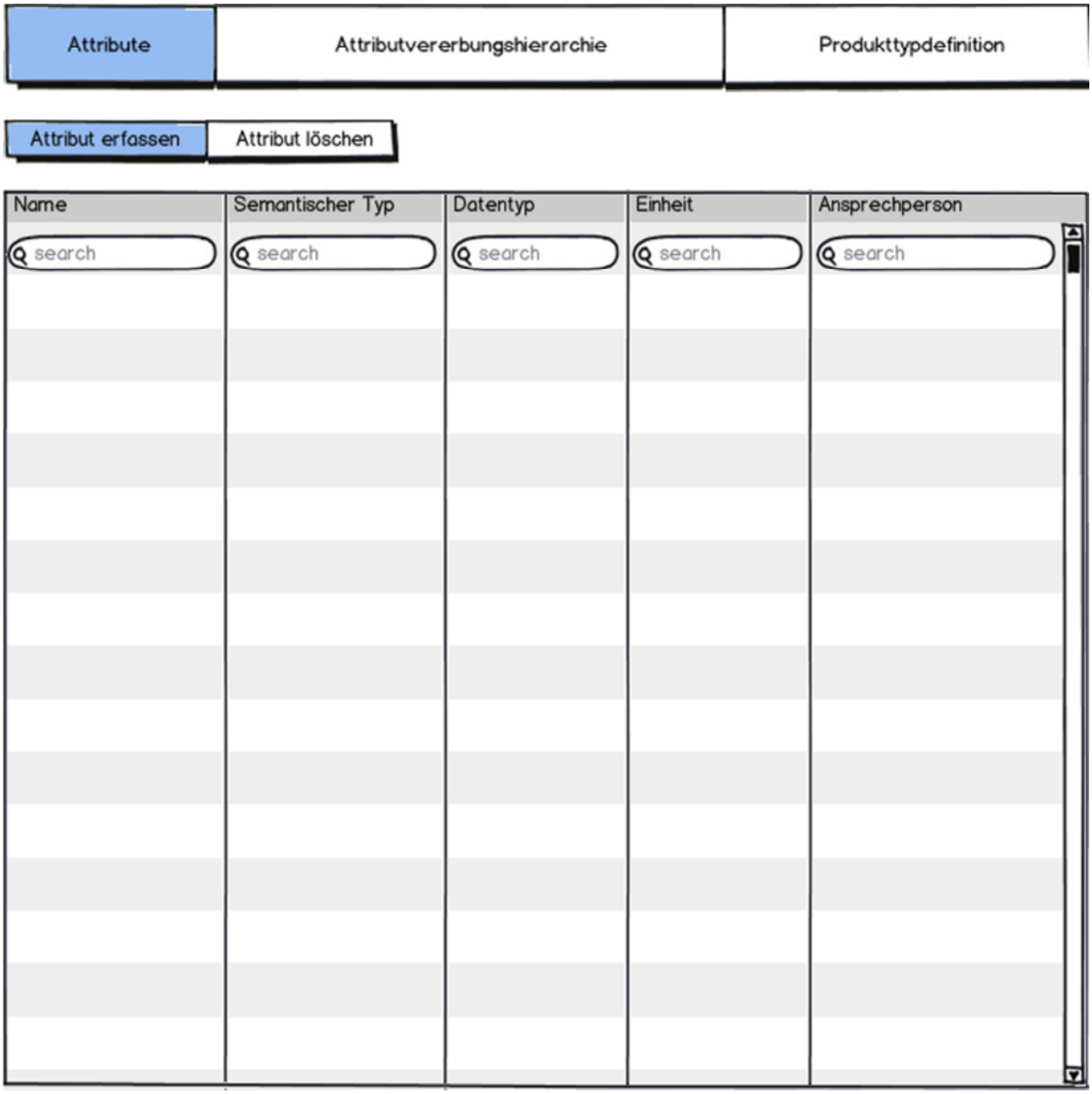


Abbildung 16 Attributliste Variante 3

4.3.3 Attributvererbungshierarchie

Abbildung 17 Attributvererbungshierarchie Variante 3

4.3.4 Attributvererbungshierarchiedetail

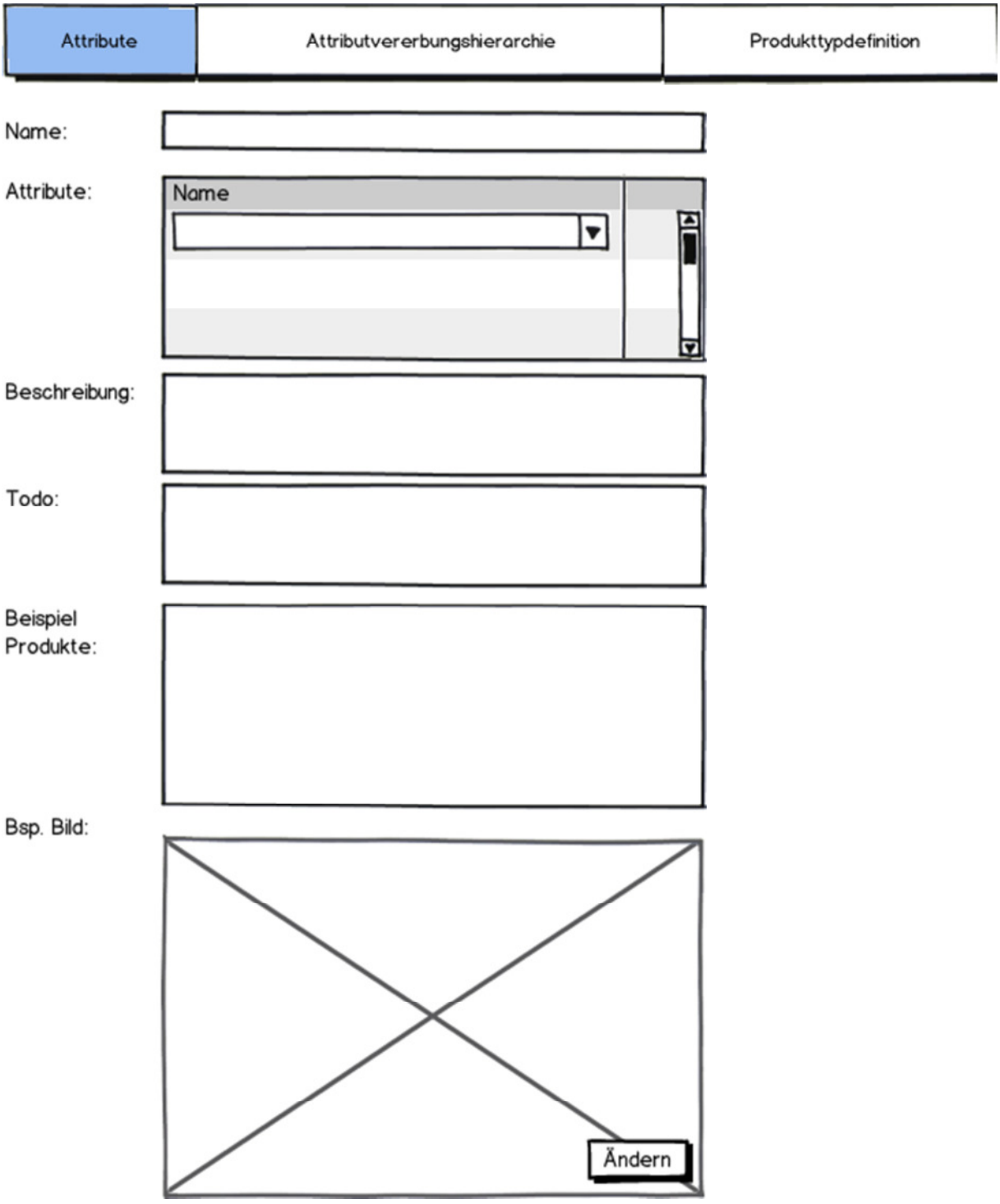


Abbildung 18 Attributvererbungshierarchiedetail Variante 3

4.3.5 Produkttypdefinition

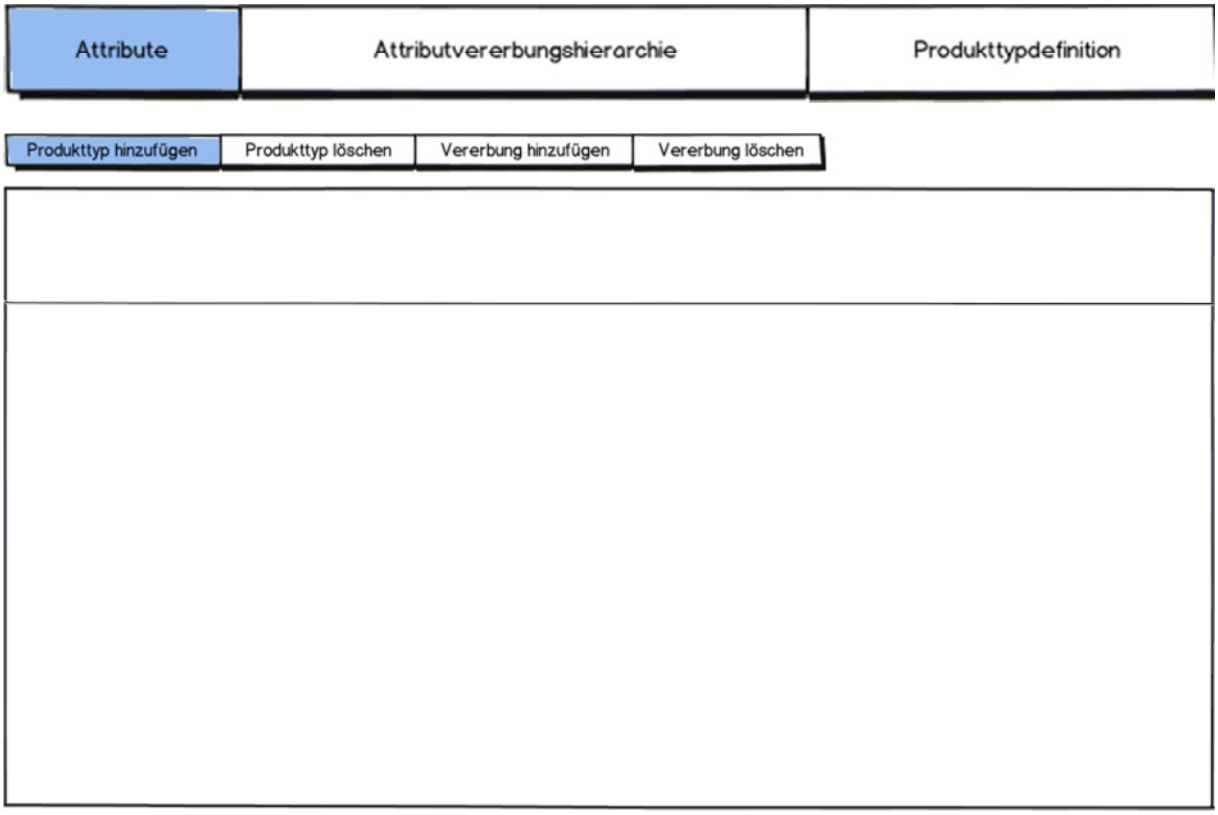


Abbildung 19 Produkttypdefinition Variante 3

4.3.6 Produkttypdefinitionsdetail

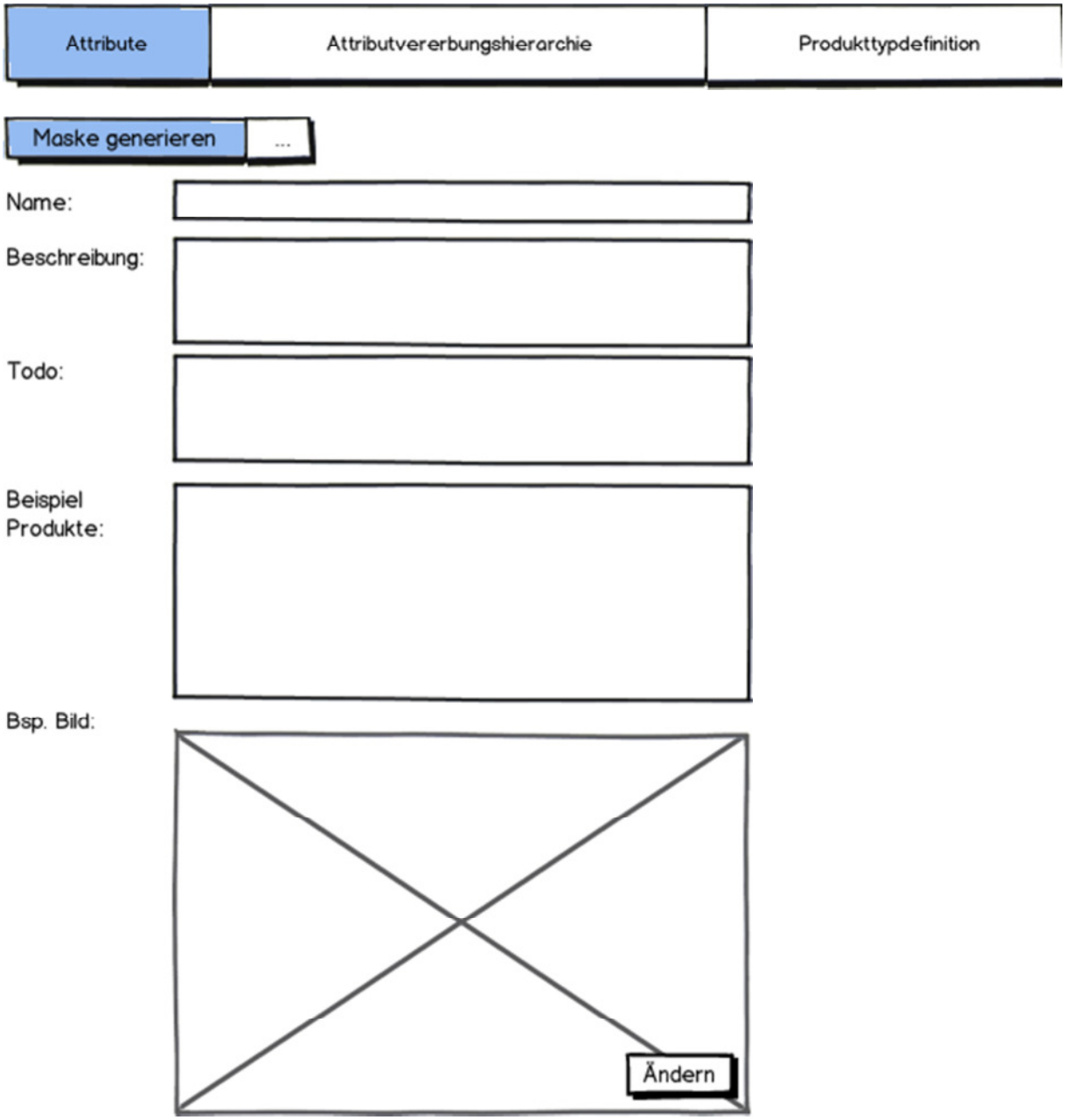


Abbildung 20 Produkttypdefinitionsdetail Version 3

4.3.7 Maskenansicht

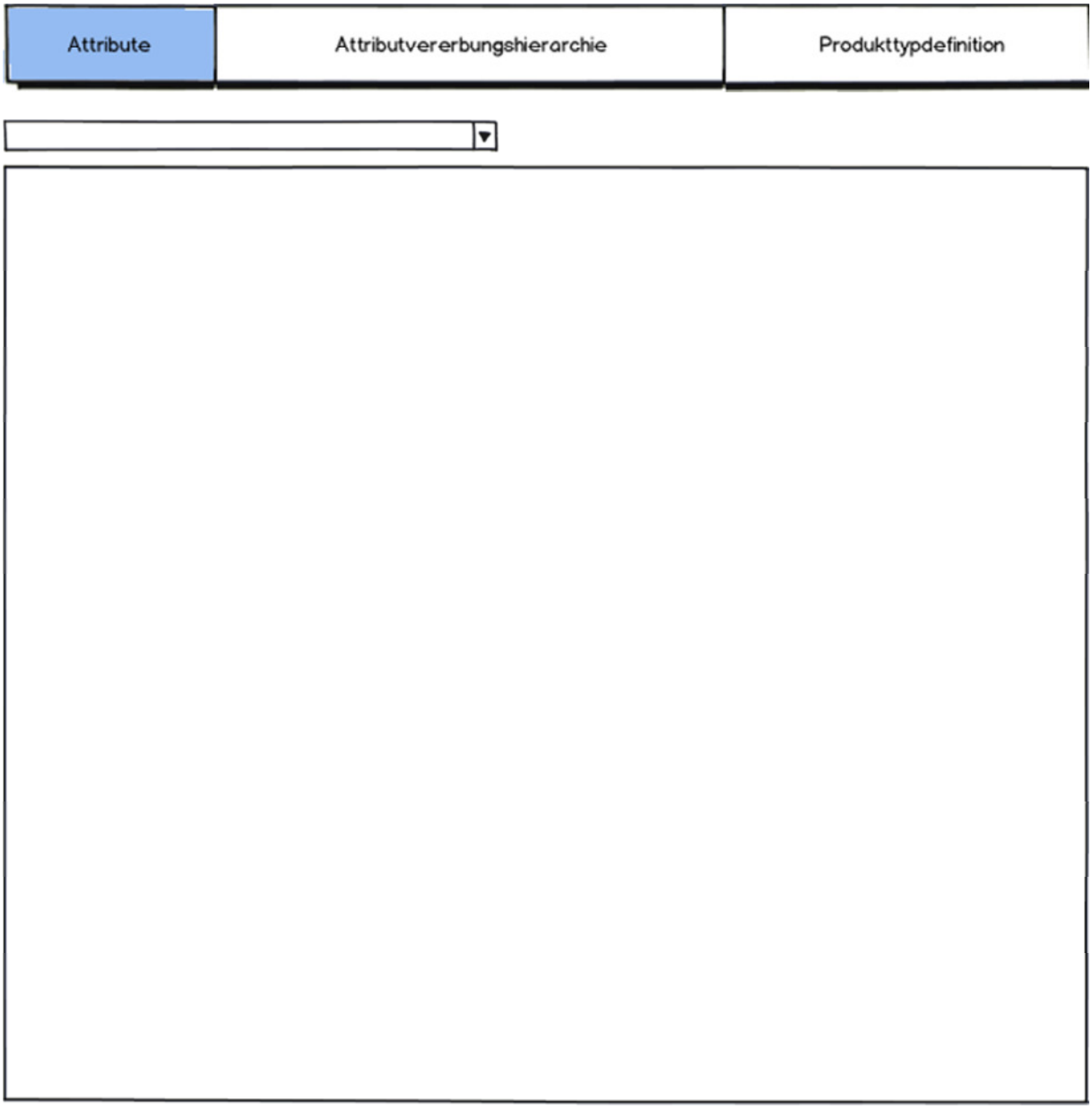


Abbildung 21 Maskenansicht Version 3

4.3.8 Testsetup

Wir haben D. Keller die Wireframes ohne Beispieldaten aufgelegt und mit einer Kamera aus der Vogelperspektive seine Aktionen aufgezeichnet.

Er bekam den Auftrag aus einem Katalog die Attribute und Metaattribute von zwei vorgegebenen Produkten zu erfassen. Ein typisches Produkt und ein eher spezielles Produkt.

Der Test fand um 17:00 Uhr statt.

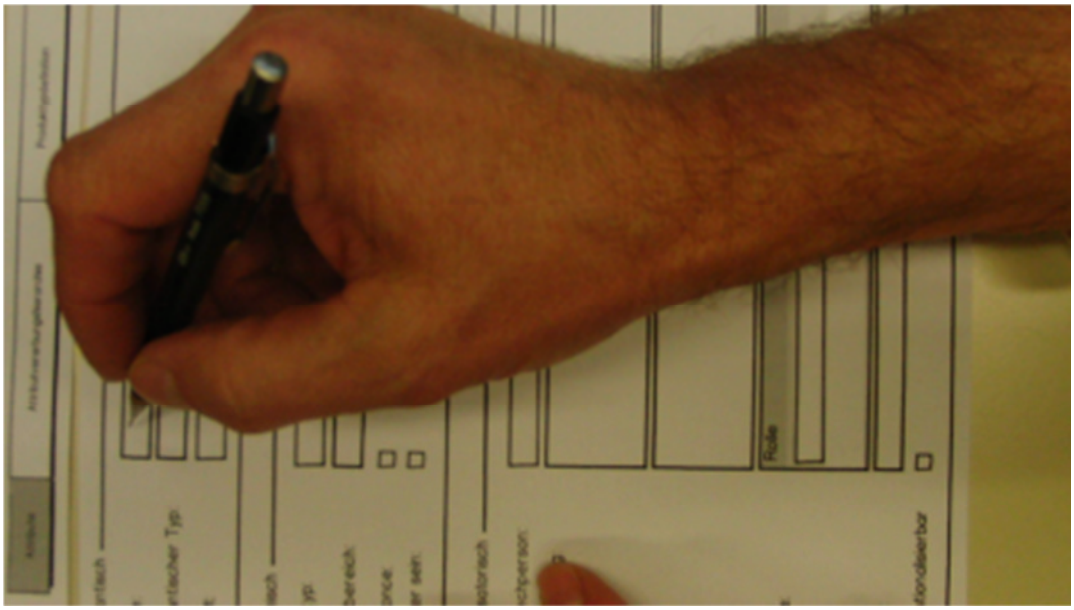


Abbildung 22 Testsetup des Papierprototyptests

4.3.9 Diskussion Papierprototyp-Test

Zeit	Beobachtung	Massnahme
01:55 04:58	Die Bedeutung des Felds „Semantischer Typ“ ist nicht bekannt. Bei 4:58 weiss der User allerdings, was er in dieses Feld einfüllen kann.	Dieses Feld ist noch nicht fertig gedacht. Wir wissen, dass in diese Richtung weitergedacht werden muss. Dies wird Thema einer späteren Iteration sein.
04:00	Der User möchte nach dem erfassen eines neuen Attributes mit einem OK Button zurück. Diesen kann er nicht finden, weil es ihn nicht gibt.	Wir haben es dem User erklärt und nachfolgend war es kein Problem mehr.
04:55	User möchte ein Attribut welches mehrere Werte hat modellieren. Mit dem Kommentar „Ich mache es einfach“ erstellt er ein Attribut „Länge mal breite“. Die Einheit wählt er Millimeter.	In der aktuellen Version gibt es noch keine explizite Unterstützung für solche Attribute. Diese Thematik wird in einer späteren Iteration angegangen.
10:48	User möchte “Ändern” Button welcher sich auf das Bild bezieht nutzen, um die Daten dieser Ansicht zu speichern.	Ändern-Button wird entfernt. Um das Bild zu ändern kann man einfach draufklicken, wie bei einer Klasse.
14:44 18:30	User bemerkt, dass er ein Attribut nun mehrfach in der Vererbungshierarchie hat.	Gut wäre, wenn das Programm bei der Detailansicht einer Klasse zeigt, von welchen Klassen geerbt wird und welche Attribute somit schon vorhanden sind. (18:30 -> dies wird explizit vom User so

		gewünscht)
15:30	User möchte zu den Attributen einen Namen angeben, unter welchem das Attribut in der Klasse erscheint.	Siehe nächster Eintrag zu 15:30.
15:30	User möchte Preis pro 100 Stück als Attribut haben.	Unsere Modellierung hätte ein Attribut "Auslieferbare Menge" gehabt; die konkrete Menge wäre dann eine Variation, also erst in der Produktivdatenbank gespeichert. Hier müssen wir prüfen, ob unser Vorschlag akzeptiert wird, oder ob wir etwas Entsprechendes im Programm einbauen müssen.
22:28	User gibt Beschreibung (Metaattribut) ein und meint, dass es sich dabei um die spätere Produktbeschreibung handelt.	• Dieses Feld sollte "Bemerkung" heissen; es ist weniger wahrscheinlich, dass es ein Bemerkungsattribut geben wird, eine Beschreibung aber ziemlich sicher. • Eine bessere optische Trennung von Attributen und Metaattributen sollte den User zusätzlich unterstützen. • In der Attributvererbungshierarchie braucht es die Möglichkeit, die Attribute der Klassen anzuzeigen. • In der Ansicht einer Klasse oder eines Produkttyps sollte die Attributvererbungshierarchie (wahlweise mit Attributen) sichtbar sein.

Neben diversen kleinen Änderungen sind vor allem zwei markante Fortschritte aus diesem Test hervorgegangen:

- Bildschirmgrösse definiert und alles auf die gleiche Grösse gebracht
- Raster eingefügt, damit alles immer am gleichen Ort zu finden ist und sicher der User besser zurechtfindet (z.B. Unterschied zwischen Attribut und Metaattribut)

4.4 Variante 4

Siehe Kapitel „2 Resultat".

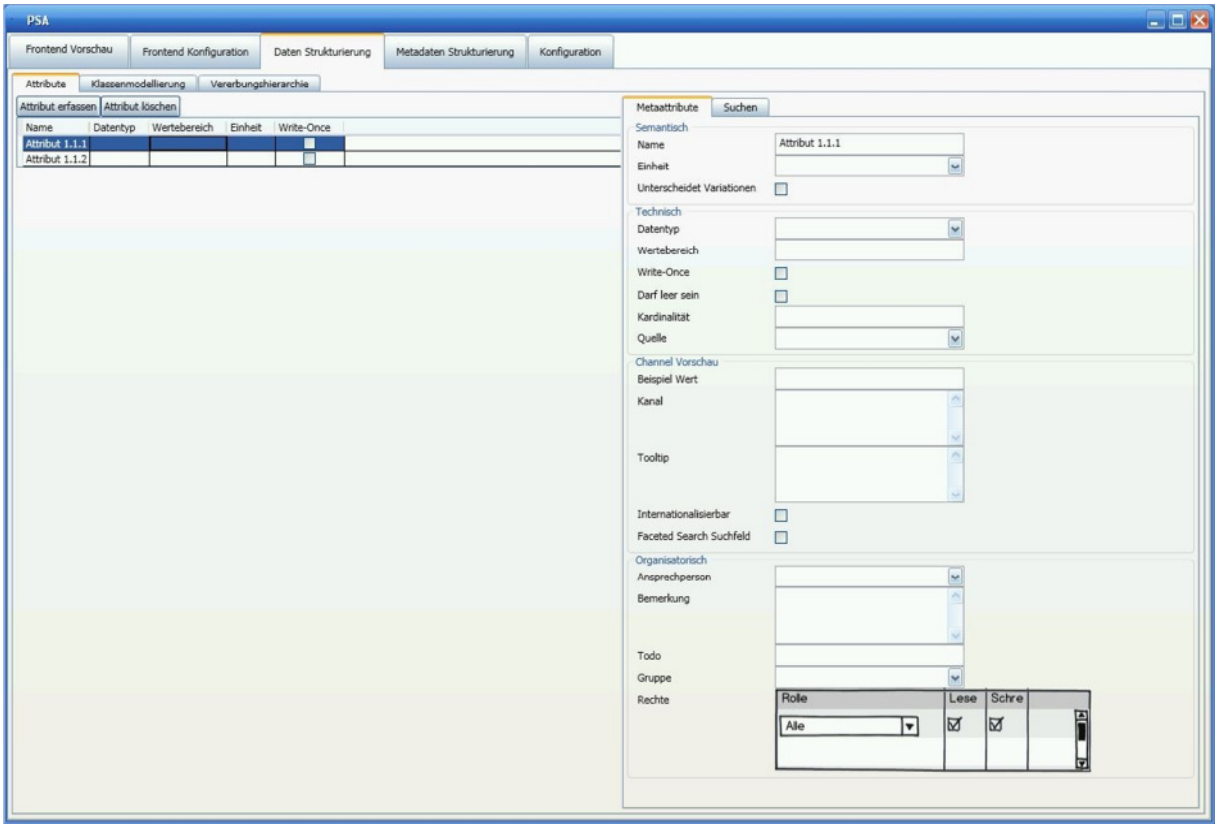
5 Nachtrag

In diesem Kapitel zeigen wir aktuelle Screenshots von unsrem Programm.

Um mit dem Kunde über unser Programm zu sprechen haben wir die noch nicht ausprogrammierten Ansichten durch Wireframes ersetzt. Dies ermöglichte uns mit dem Kunde über die aktuellen Features zu sprechen und nicht über „was man noch alles machen könnte“.

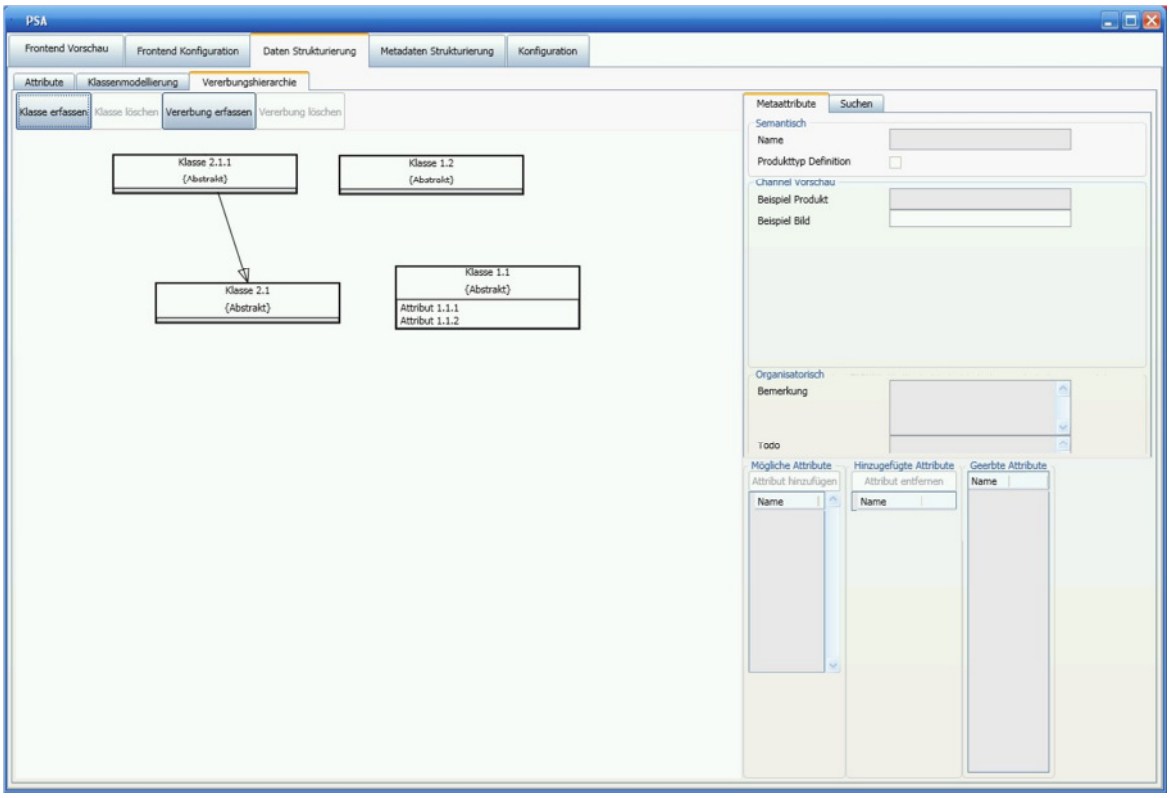
5.1 Attribute

Einer unserer ersten Ansichten ist die Attributansicht. Sie entspricht praktisch eins zu eins unserem geplanten Wireframe. Die Metaattribute „Rechte“ haben wir noch nicht umgesetzt. Um zu zeigen, dass wir dies aber geplant haben, haben wir sie als Wireframe in unser laufendes Programm eingefügt.



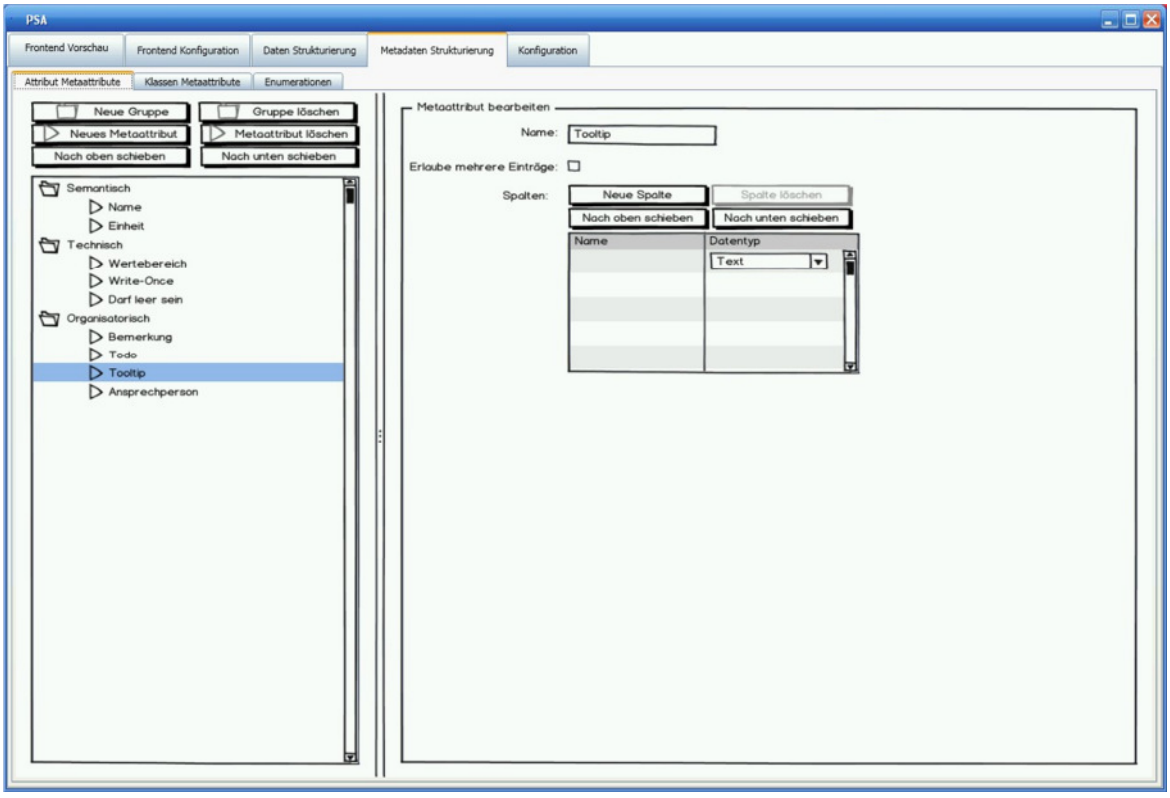
5.2 Vererbungshierarchie

Auch die Vererbungshierarchie entspricht unserem Wireframe.



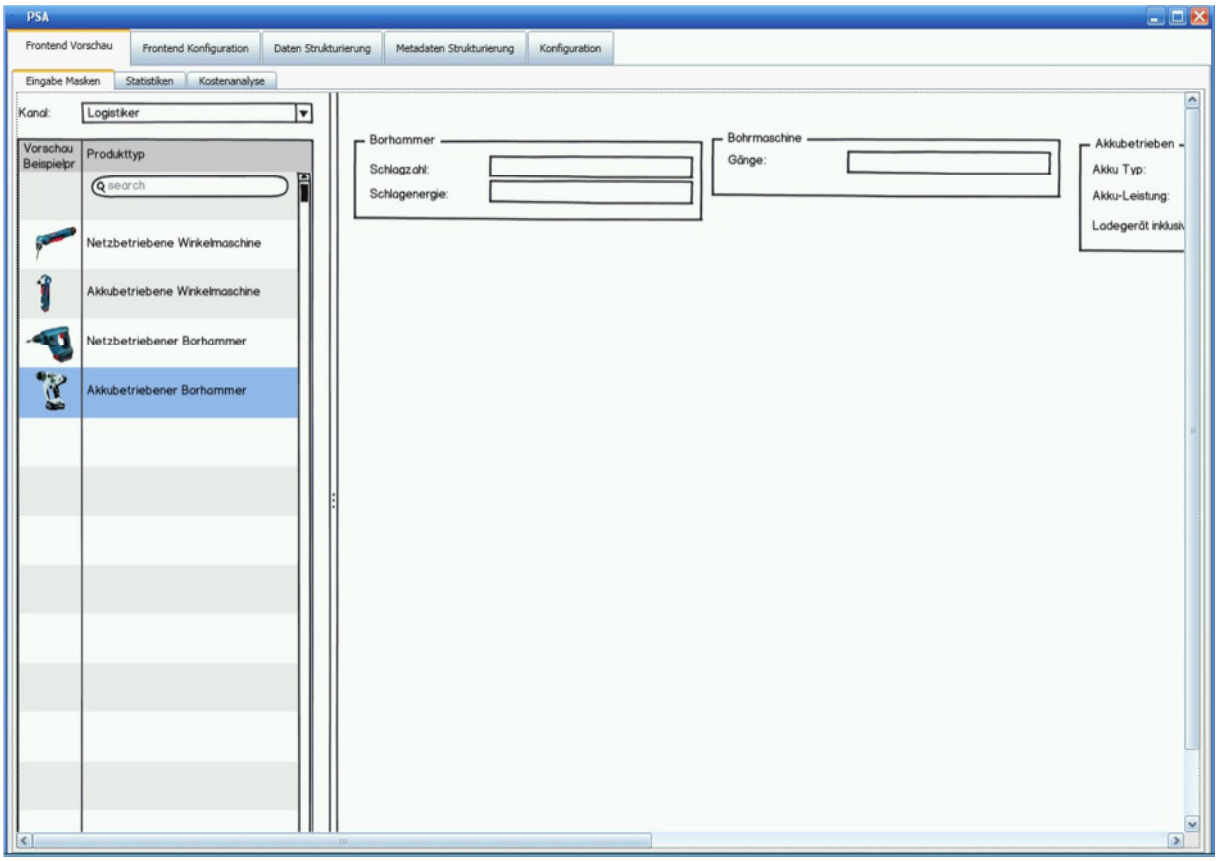
5.3 Metaattribut Strukturierung

Diese Ansicht wird komplett durch ein Wireframe repräsentiert.



5.4 Frontend Vorschau

Auch diese Ansicht ist zurzeit nur als Wireframe vorhanden. Dieses ist sogar interaktiv: Mit einem Mausklick kann man sich die Masken der verschiedenen Produkttypen ansehen.



6 Reflektion

Wireframes

Trotz anfänglicher Ablehnung mussten wir feststellen, dass sich die Wireframes sehr lohnen. Sie haben uns die Kommunikation mit dem Kunden ziemlich erleichtert. Auch die Kommunikation im Team wurde einfacher: Wir mussten unsere Ideen konkret niederschreiben und konnten so Detailprobleme, welche in den Gedanken schnell mal untergehen, erkennen. Der Aufwand besteht nicht im Zeichnen der Wireframes sondern vor allem in deren Planung. Diese Zeit muss man aber so oder so investieren, spätestens dann wenn man ein GUI entwickelt. So konnten wir unsere Ideen immer wieder dem Kunden präsentieren und ohne viel Aufwand die Wireframes anpassen. Für die spätere Implementierung bieten sie den Vorteil, dass man erstens weiss was alles auf einem zukommt und so besser planen kann und zweitens muss man sich keine Gedanken mehr zum GUI machen, sondern kann es einfach umsetzen.

Wireframes im laufenden Programm

Da wir so gute Erfahrungen mit den Wireframes gemacht haben und wir eine Kundenpräsentation vor uns hatten, entschieden wir uns das ganze Programm und alle Features zu planen. Damit wir dem Kunden all unsere Ideen präsentieren konnten, auch jene, welche wir noch nicht programmiert hatten, entschlossen wir uns die Wireframes in unser laufendes Programm einzubinden.

Szenarios

Die Szenarien halfen uns einige Detailprobleme wie z.B. dass Synonyme für Attribute nur wenig Sinn ergäben zu erkennen und lösen. Wir haben alle Szenarios mit dem Kunden validiert. Da unser Ansprechpartner zu diesem Zeitpunkt noch keine reale Praxis Erfahrung hatte, war es relativ schwierig detaillierte Szenarien zu erfahren. Dementsprechend kam auch wenig Feedback zu unseren Szenarien.

Papierprototyptest

Der Test war sehr Aufwändig, aber eine gute Erfahrung. Wir konnten auch zwei grössere Probleme identifizieren und diesen entgegenwirken. Unteranderem war dem Benutzer nicht klar, auf welcher Ansicht er sich gerade befindet.

Einen nächsten Test würden wir nicht mehr am Abend durchführen: Es ist für alle Beteiligten sehr viel Aufmerksamkeit erforderlich. Zum einen für den User, welcher sehr viel abstrahieren muss und ein unbekanntes Programm bedienen muss, zum anderen für die Test-Durchführer, die im Hintergrund Daten generieren müssen und auf unvorhergesehene Bedienung des Programms achten müssen.

7 Verzeichnisse

7.1 Abkürzungsverzeichnis

Fyayc	Foryouandyourcustomers
UInt2	User Interface 2 (Modul an der HSR)
PSA	Produkt Struktur Architekt

7.2 Abbildungsverzeichnis